IS 2900 – Project on IT Applications

Final Report

School Management System

Group Kripto

Index No	Name
175053T	Nasrina M.I.F
175064D	Prasandika K.A.C
175083J	Thilakarthna W.M.D.S
175090D	Welagedara J.M

Supervised by:

Mr. S.M.U. Premasiri
Ms. W.A.S.N. Wijethunge

Client:
Arthur C Clark Institute for Modern Technologies
Katubedda, Moratuwa

Faculty of Information Technology
University of Moratuwa

2022

Declaration

We declare that this report is our own work and has not been submitted in any form for another degree or diploma at any university or other institution of tertiary education. Information derived from the published or unpublished work of others has been acknowledged in the text and a list of references is given.

Names of Students	**Signatures of Students**
175053T - Nasrina M.I.F	
175064D - Prasandika K.A.C	
	
175083J - Thilakarathna W.M.D.S.	
175090D – Welagedara J.M	. 

Date: 18/06/2022

Supervised by:

Names of Supervisors	**Signatures of Supervisors**
S.M.U.Premasiri Date:	
W.A.S.N. Wijethunge Date:	

Abstract

In the current world, parents do not have enough time to pay attention to their children's school activities and studies which is very important for their future. Though there are learning management systems like My class campus, fedena, Moodle etc. to support this issue, there are several disadvantages in them basically because those are based on monolithic architecture. Therefore, through the proposed system a web-based school management system will be developed to manage the essential activities in a school learning system more efficiently using MVC architecture. Basically students, parents, teachers and administration who are the users of the system, are able to observe the progress of students. EJS, HTML, Java Script, CSS and Node JS are used in developing the frontend web and backend of the application respectively. Therefore, through the proposed system the drawbacks in the current school management systems will be avoided while establishing a strong relationship among students, parents, teachers and school administration.

Table of Contents

List of Figures/Tables

Chapter 1

Introduction

1.1 Introduction

The education of children is a significant factor in the life of them. But in the modern world individuals in each and every society are spending a very busy life style. As a result, parents do not have enough time to pay attention to children's progress in studies. Therefore, in order to strengthen the relationship between the parents, students, teachers, school administration and to facilitate paperless administration of schools, a school management system is proposed as a solution. There are many learning management systems which have already been developed. But in those systems parents are not included as a user. Therefore, in the proposed school management system parents, students, teachers and administration are allowed to access the system. The proposed system will handle only the administration of the educational activities of students. The aim of the system is to develop a web-based system to manage the essential activities in a school learning system more efficiently and effectively. The objectives of developing this system are, developing a system which can be used to observe the progress of a student, teachers and parents, automate the administrative activities which are currently done manually. The rest of the paper is organized as follows. Section 2 reviews the related literature. Section 3 describes the technology adapted. Section 4 discusses the approach. Section 5 includes analysis and design. Section 6 is about the implementation. Section 7 is on evaluation of the system.

1.2 Problem in Brief

Nowadays school progress meetings are getting quarterly. Therefore, we can see the gap between teachers and parents. Then it has limited time to discuss the progress of the student with the parent. As well as parents cannot get continuous progress about the student when it happens quarterly. Within this time there can be lot of changes of that particular student and sometimes there can miss or do not remind the special things of student, since there is a huge gap between two quarters. Therefore, parents cannot get a true picture about students continuously. Students may leave home to school. But he has not attended to the class and does not do activities within the class. Then that also a big problem. As usual if the student does it, that is badly affected to the student. Parents could not view the daily attendance of the student. When it is dealing with manual progress reports, sometimes students change their results if they have got low marks. That is a major problem today.

Most of the time parents cannot see the students' creative works except exhibition when they doing in school times. So, it is better to have displayed these works through a dashboard. Then parents can see students' works and motivate them more and more to do creative activities. Going to implement a School Management System by addressing above mentioned problems.

1.3 Aim and Objectives

1.3.1. Aim

Developing a web-based system which can be used to observe the clear progress of a student and automate the administrative activities which are currently done manually.

1.3.2. Objectives

- To develop user registration and login service
- To develop student profile
- To develop dashboard
- To develop report generation service
- To develop user management service

1.4 Proposed Solution

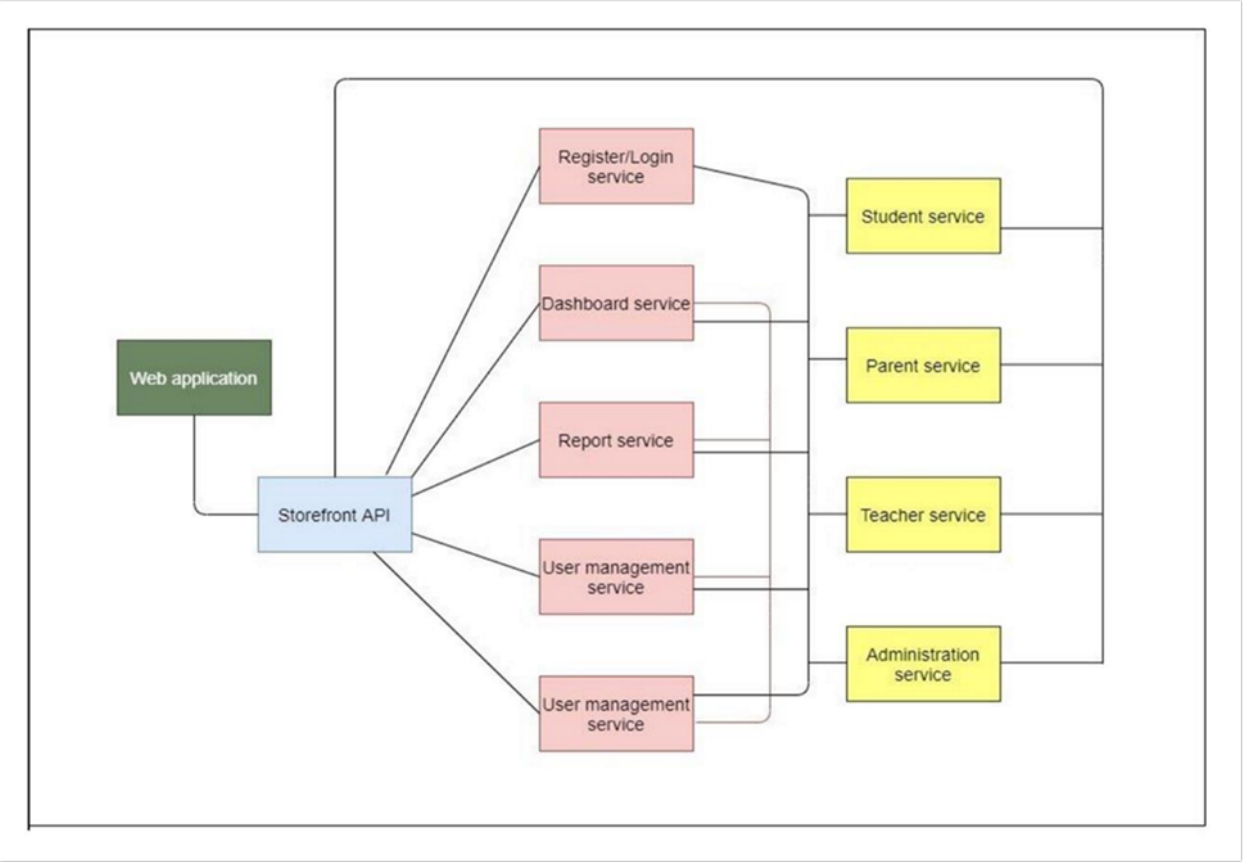


Figure 1: Proposed Solution

Registration Service: - When student, teacher or an administration member is enrolled here we keep records like their admission number, name, family member details and other details are recorded. Parent also get access to this system with the registration of the student.

Report Service: - System will auto generates the reports to analyze student's performance and progress. After generating these reports, it will able to compare the reports with each student and each term. There will be term wise exam result reports and assignment marks reports for each student. They can be viewed only by the relevant student, parent and the teacher. Attendance reports of each student will be generated monthly. Term wise marks of each subject and average marks of each subject of each class can be viewed as charts. These charts can be viewed only by the relevant teacher. So, using these charts teacher can identify the places that should be change for the betterment of the class.

Dashboard and notice board Service: - In a nutshell, this is a notice board. Students can upload their creative works to the dashboard. Admins and teachers can upload special notices in the noticeboard. At the time it is uploaded no one can view what has uploaded to the dashboard. It appears on the dashboard after the article is approved by an admin. Achievements that are acquired also appear on this. This can be seen by every party in this system.

User Management Service: -Manages user login for parents, staff, administrators and students with customizable role privileges. Assigns privileges for predefined roles to carry out relevant operations.

Student service: - Compare their own exam, assignment marks, attendance records, achievements via reports. Students can know about the home work and upcoming events in the school.

Parent service: - Parents can access, the system and view their children's attendance and reports at the same time. Special events in the school can be viewed via dashboard.

Teacher service: - In the teacher service, a teacher will upload assignments, quizzes and exam papers. After marking the papers marks will be uploaded by the teacher. At the same time teacher will mark the attendance of the students from his or her mobile phone.

Administration service: -When a new student, teacher or a parent is registered it will be done by the administrative service. After a teacher upload exam results and students reports, it will be checked and make them visible in the dashboard by the administrative service.

1.5 Resource requirements

- Computer with 4GB RAM, 2.4GHz or more processing power
- Backend: NodeJS
- Frontend: JavaScript,React, EJS, HTML, CSS
- Database: MySQL

1.6 Summary

This chapter provide a brief description about our project. Our main goal is to use microservices architecture to overcome the disadvantages of monolithic architecture.

In chapter 2 it describes about other school management systems.

In chapter 3 it describes about the technology we use.

In chapter 4 it describes about the approach for the problem domain.

Chapter 5 is about the analysis and design of the system.

Chapter 6 is on the implementation of the system

Chapter 7 is on test results of the system

Chapter 2

Literature Review

2.1 Introduction

There are many open-source free and paid school management systems. Each of them performs different tasks in a school.

2.2 Existing Systems

2.2.1 MyClassCampus

MyClassCampus is a leading platform to provide school management. Also providing facilitate to conversion between students, teachers, and parents for incredible correspondence and general productivity. Designed with advanced features to save the time of admin and gives access to vital information on a single click. The thing is there is no specific facilities to school activities.

2.2.2 Fedena

Fedena is an open-source school administration software that largely focuses on handling records.

Figure 2: Fedena school administration software

2.2.3. School Tonic

School Tonic is web-based school management software which designed to facilitate paperless administration of schools. It has the ability to maintain academic history of students, staffs, various records, etc.

2.2.4. Moodle

Moodle is a free and open-source learning management system written in PHP and distributed under the GNU General Public License. This allows for extending and tailoring learning environments using community sourced plugins.

Figure 3: Moodle

2.3 Summary

There are many more school management software which were designed based on the customer specific requirements.

Chapter 3

Our Approach

3.1 Introduction

This project is designed to facilitate paperless administration of schools. School management software has the ability to host modules that allow a user to maintain academic, history of students, records, etc. Each student, teacher, parent and administrators can register to the system comes in mobile application and web interface which is implemented by react native and react.

3.2 Users

This system is used by students, teachers, parents and admins.

4.3 Inputs

User details, notes, examination, marks, marking scheme, attendance records, articles, news, student answers

3.4 outputs

Progress report, attendance report

3.5 process

Students can enroll to the system under the supervision of admin. Teachers, parents and admins are also given a username and password, so they also become users of this system. Teachers upload student activities (marks, notes, examination, announcements, marking schemes, attendance).

Then students are able to see them. Parents are able to view their child's attendance reports and progress reports which are created by the system automatically. Students can upload their assignments then teachers can correct them and release marks. Students can upload their articles. Teachers and admins can upload news. Confirmation for the articles and news is given by the admin and admin is the main role for user registration. All users can view articles and news.

3.6 Technology

School management system consists of a web-based application. For the frontend we used EJS , JavaScript, HTML and CSS.For database we used SQL.

3.6.1 EJS

EJS simply stands for Embedded Javascript. It is a simple templating language/engine that lets its user generate HTML with plain javascript. EJS is mostly useful whenever you have to output HTML with a lot of javascript.EJS is a tool for generating web pages that can include dynamic data and can share templated pieces with other web pages (such as common headers/footers). It is not a front-end framework.

3.6.2. JavaScript

JavaScript, often abbreviated as JS, is a programming language that is one of the core technologies of the World Wide Web, alongside HTML and CSS. Over 97% of websites use JavaScript on the client-side for web page behavior, often incorporating third-party libraries.

3.6.3. HTML and CSS

The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets and scripting languages such as JavaScript.

Cascading Style Sheets is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

3.6.4. Node JS

Node. js is primarily used for non-blocking, event-driven servers, due to its single-threaded nature. It's used for traditional web sites and back-end API services, but was designed with real-time, push-based architectures in mind.Node.js is an open source server environment; Node.js is free; Node.js runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)

3.7 Summary

In this chapter we described about the input, output, process and technology that we are going to use in our system.

Chapter 4

Analysis and Design

4.1 Introduction

After identifying the problem, we gathered the requirements from our client and started developing a solution. In order to gain a clear idea about the system, we designed our system using UML diagrams. We designed both the structural and behavioral diagrams.

4.2 Analysis

Before designing the system, we cleared out the user requirements that we address from our project.

Functional requirements

- All users should be able to login
- All users should be able to edit profile
- All users should be able to view notices
- All users should be able to view progress reports
- All users should be able to view attendance reports
- All users should be able to view articles
- All users should be able to view timetable
- Student should be able to view homework and assignments
- Student should be able to upload homework and assignments
- Student should be able to view notices
- Student should be able to publish creative activities
- Student should be able to View Assignment marks
- Student should be able to View timetable
- Student should be able to view upcoming events in the school
- Student should be able to access the quiz
- Student should be able to view dashboard.
- Parent should be able to view timetable
- Parent should be able to view marks
- Parent should be able to view dashboard.

- Parent should be able to view creative activities of all students
- Parent should be able to donate
- Parent should be able to pay school fees
- Parent should be able to chat with teacher
- Teacher should be able to mark and upload attendance
- Teacher should be able to upload assignments and homework
- Teacher should be able to upload notices
- Teacher should be able to view students' assignments and homework
- Teacher should be able to upload quizzes
- Teacher should be able to upload marks

Non-functional requirements

- User interfaces should be user-friendly
- Less response times
- Security and privacy

5.3 Design

Using the requirements we gathered, we drew the UML diagrams to indicate our proposed solution.

5.3.1 ER diagram

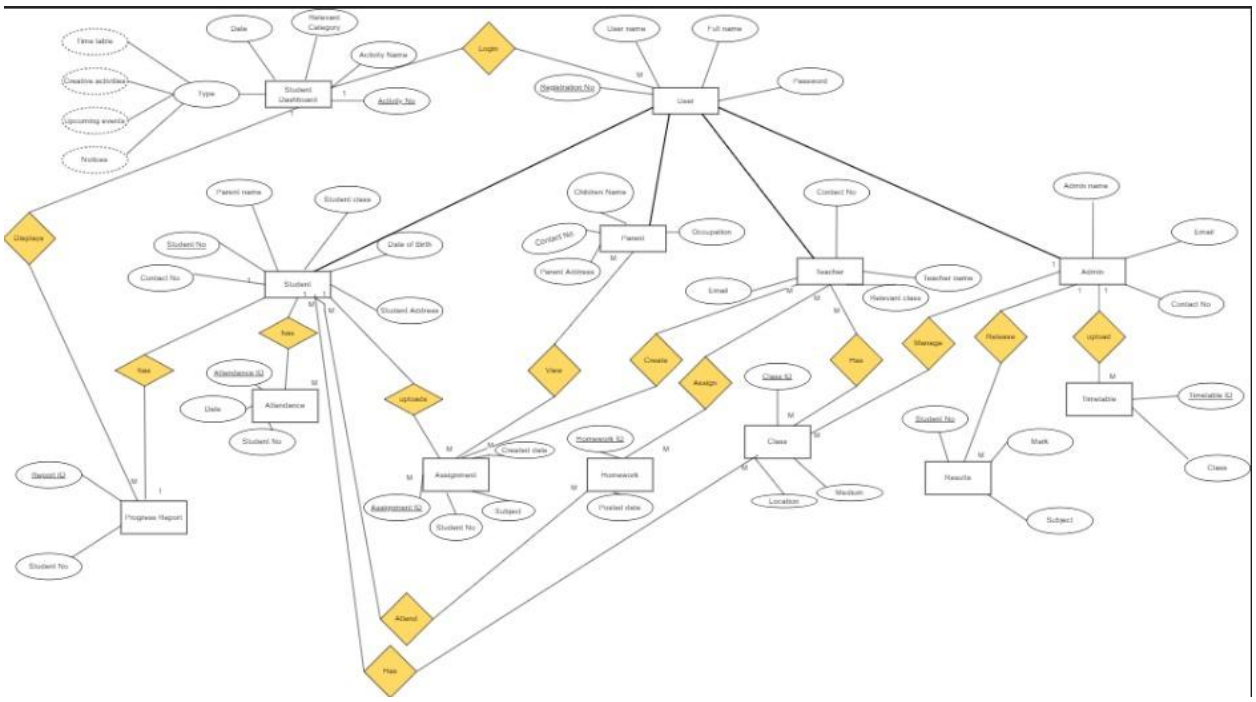


Figure 4: EER Diagram

5.3.2 System diagram

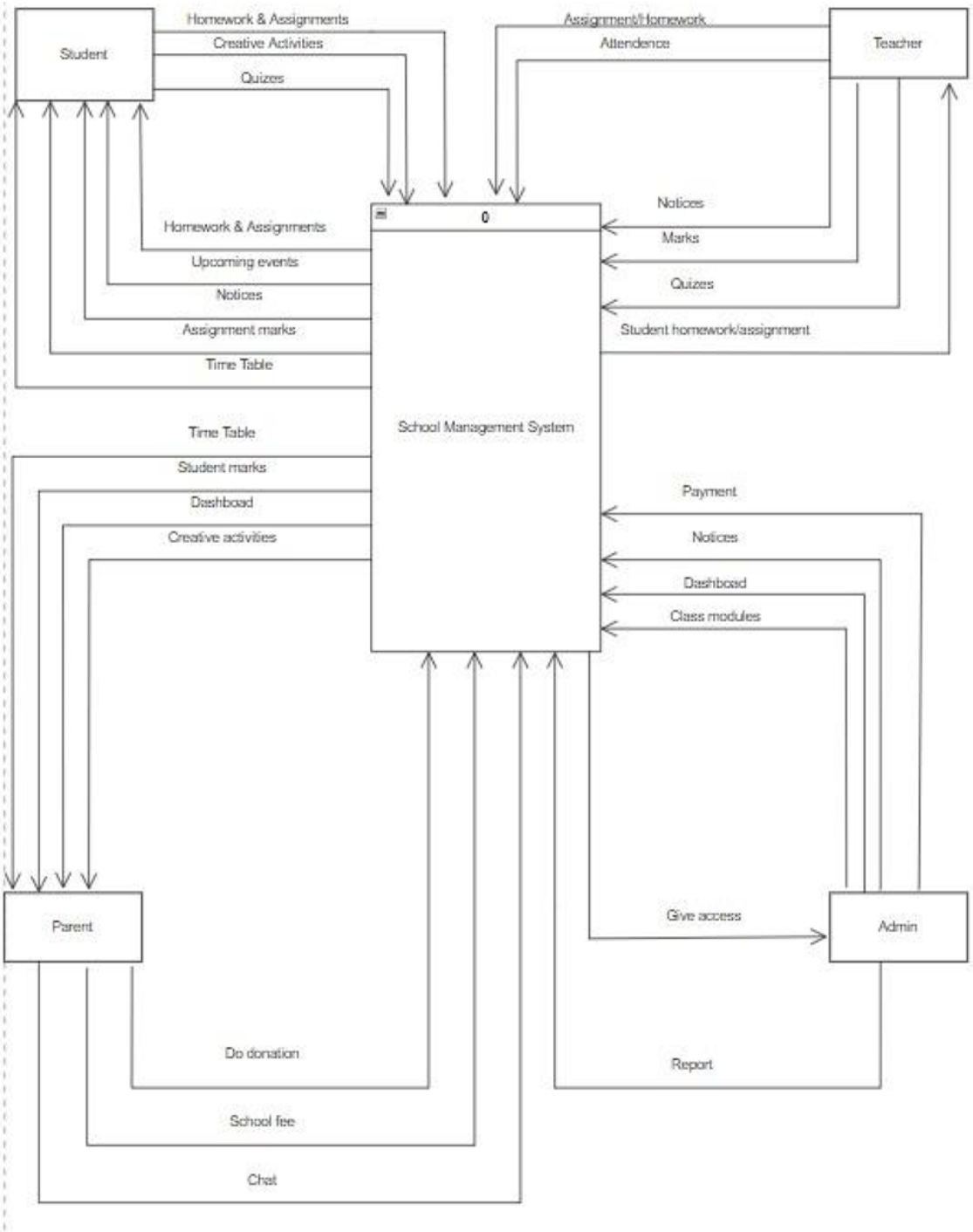


Figure 5: System diagram

5.3.3 Use case diagram

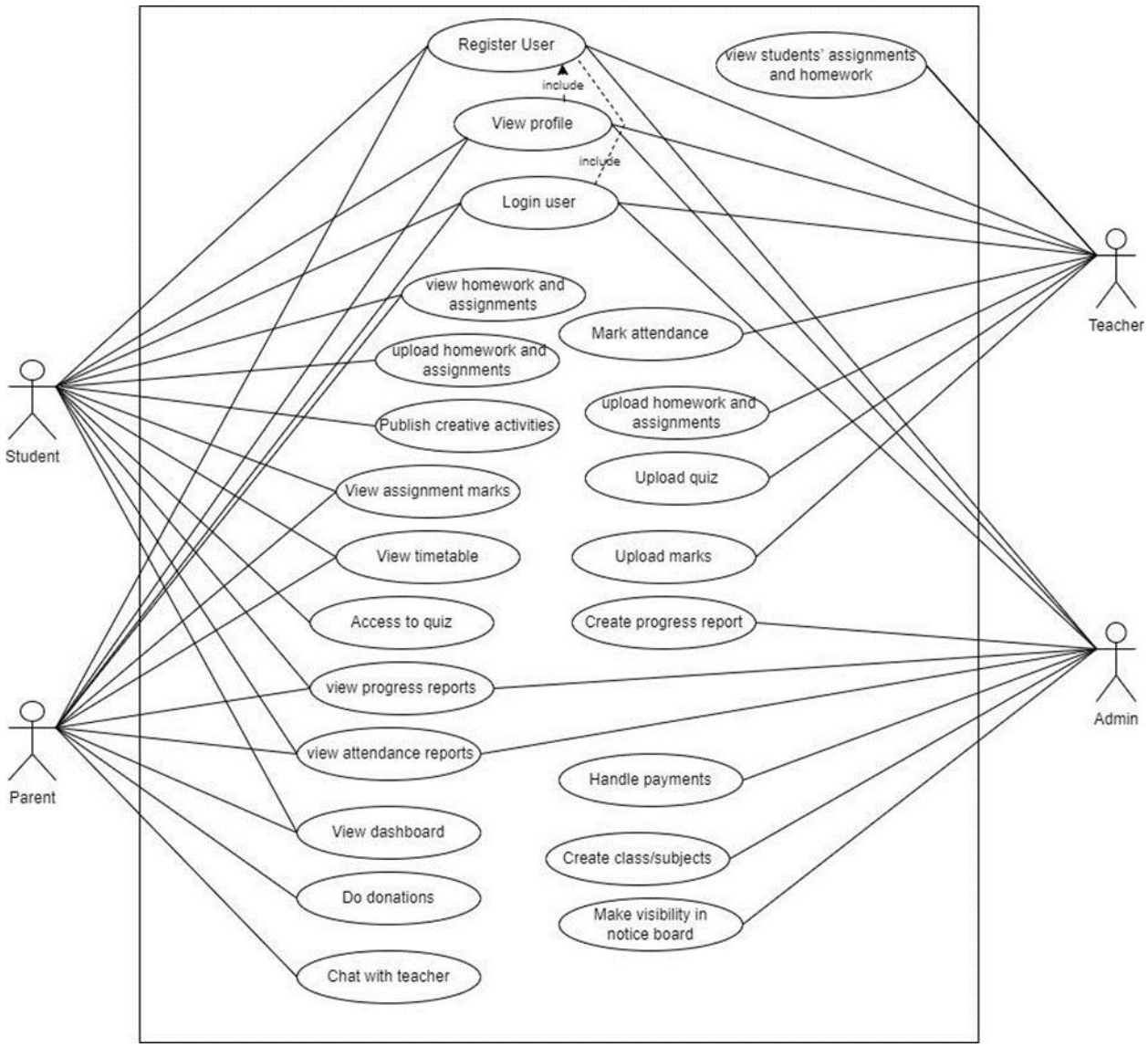


Figure 6: Use case diagram

5.3.4 Activity diagrams

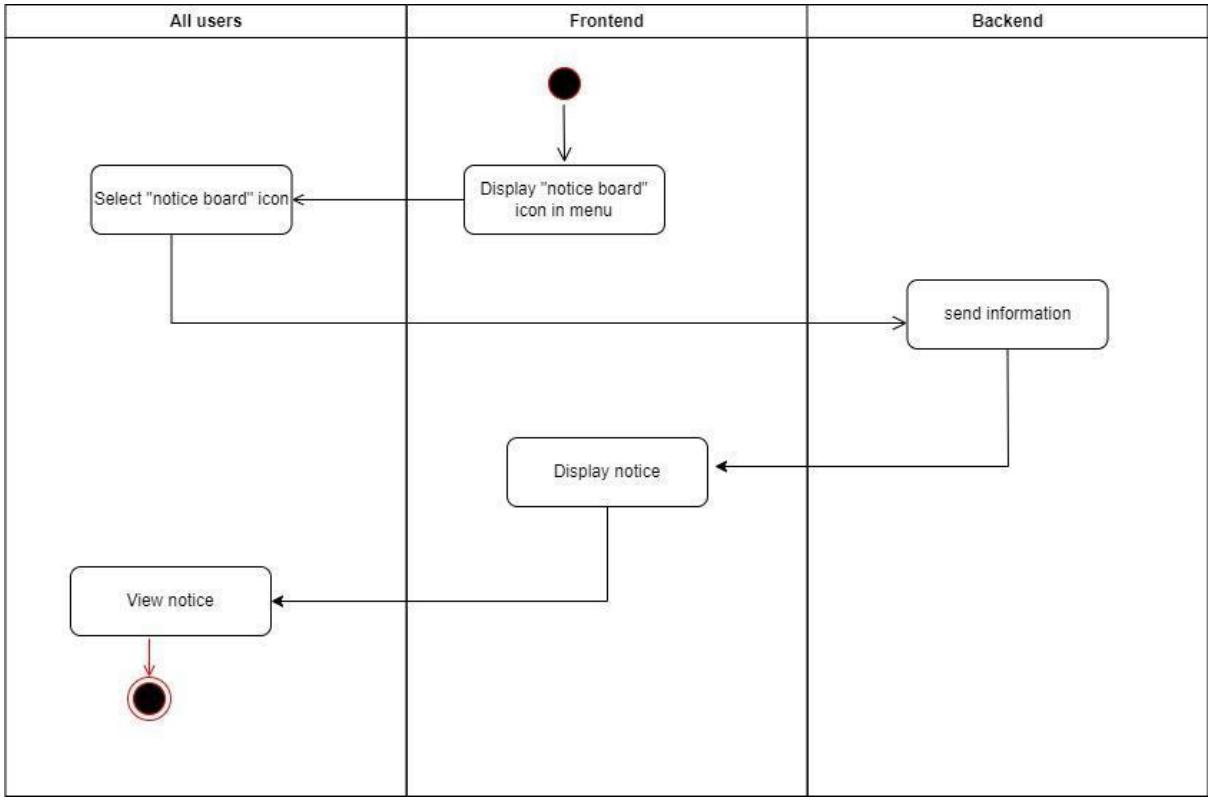


Figure 7: Activity diagram for all users

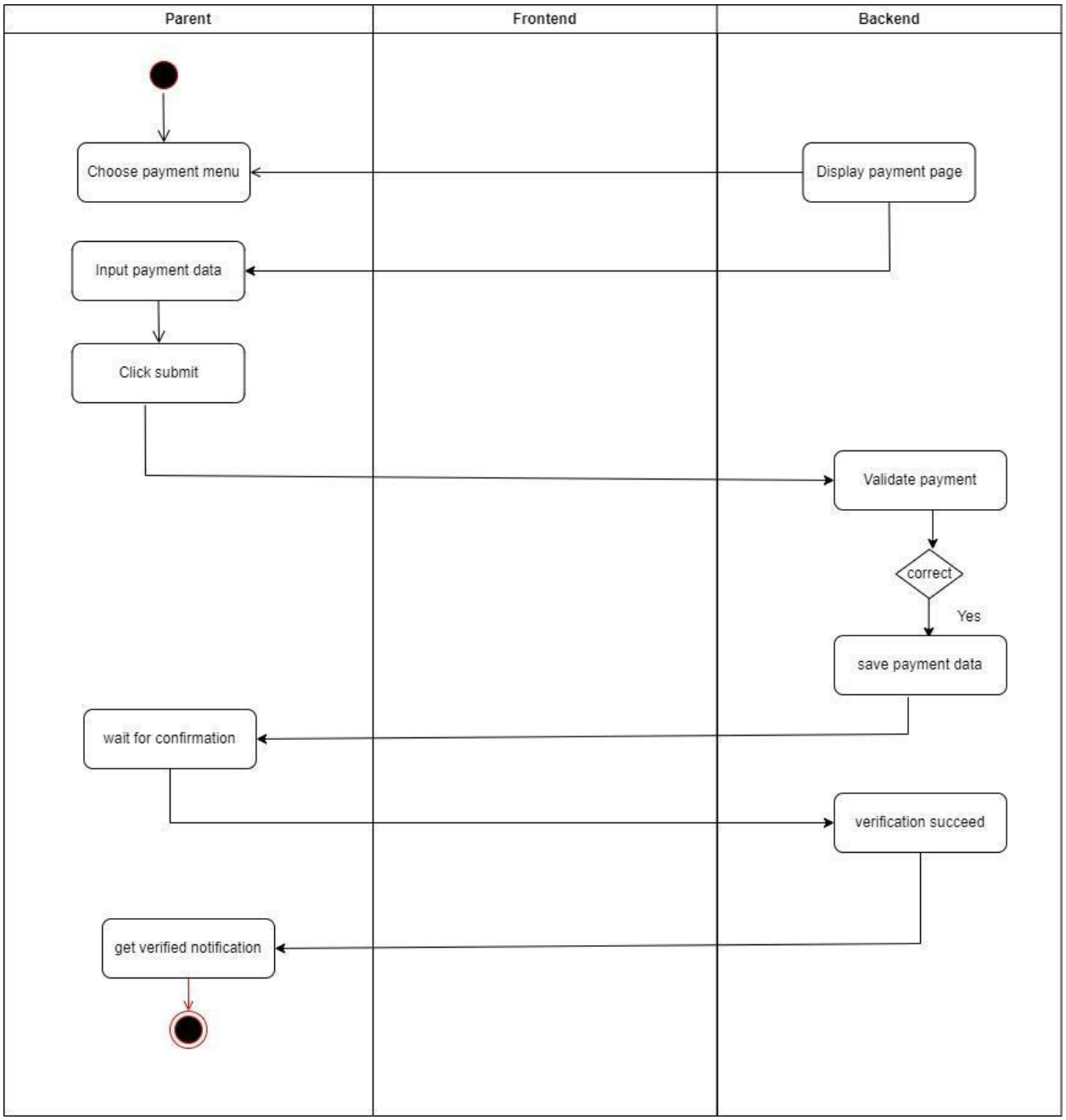


Figure 8: Activity diagram for parent to do payment

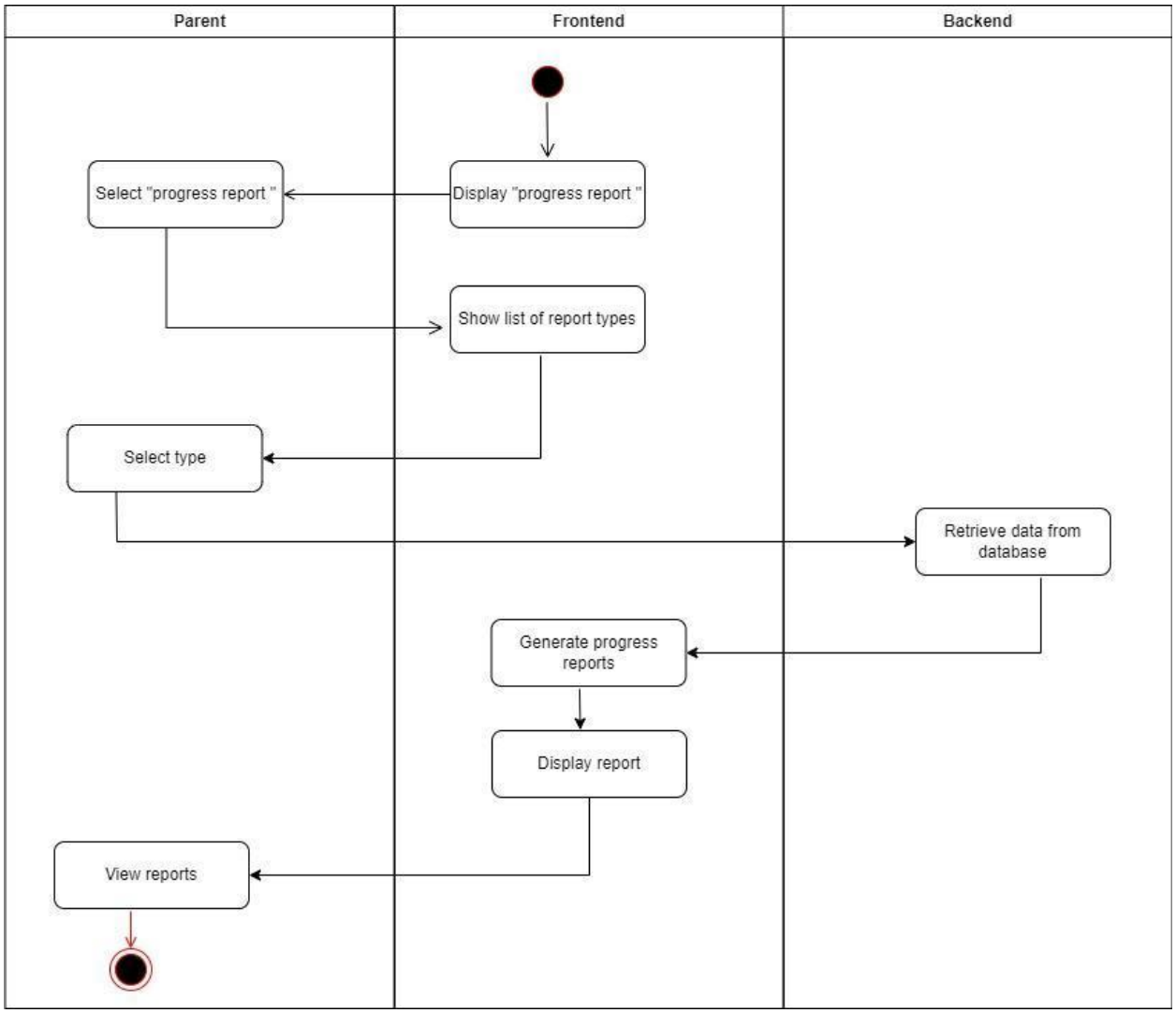


Figure 9: Activity diagram for parent to view progress reports of students

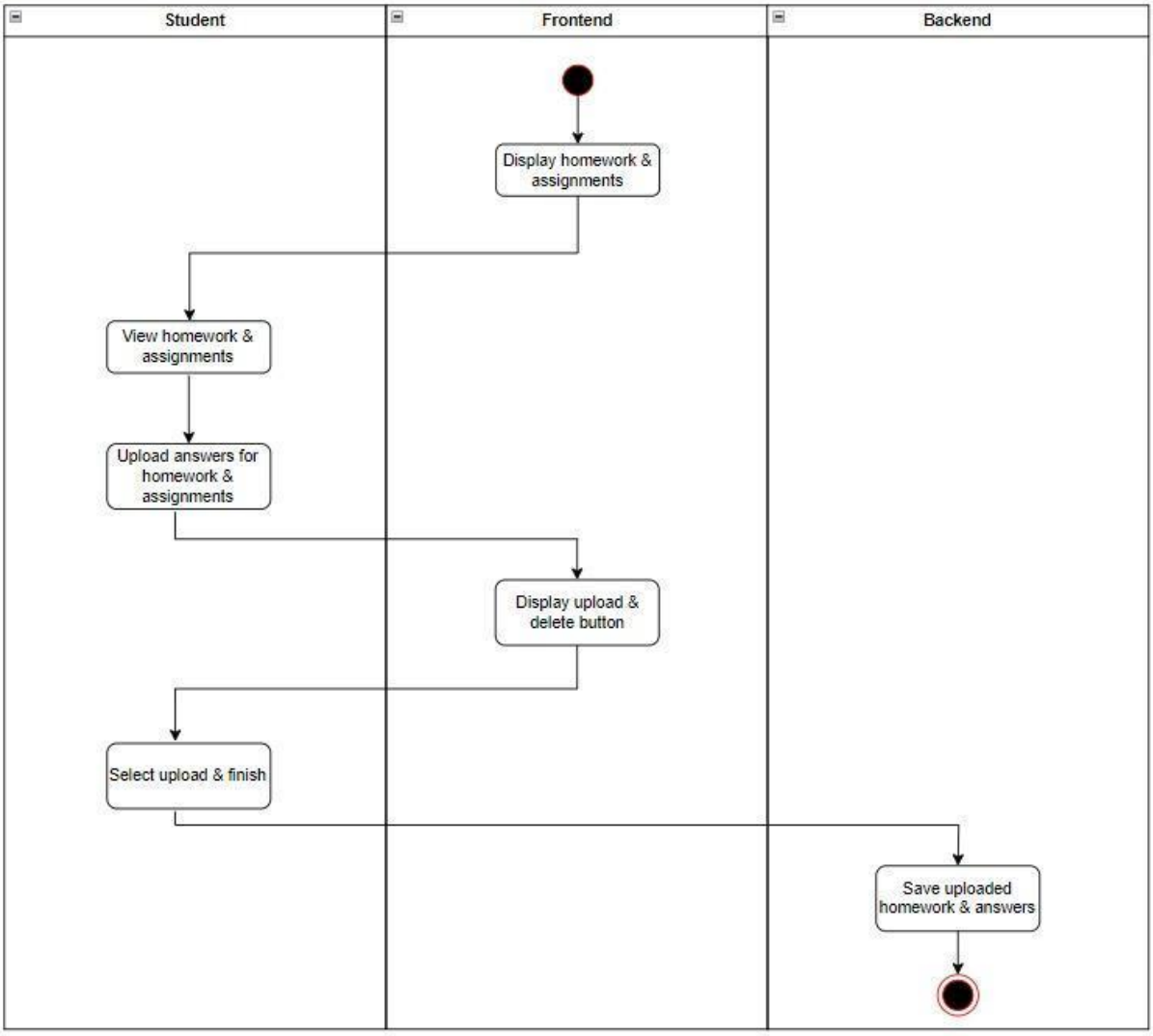


Figure 10: Activity diagram for Student to view homework & assignments

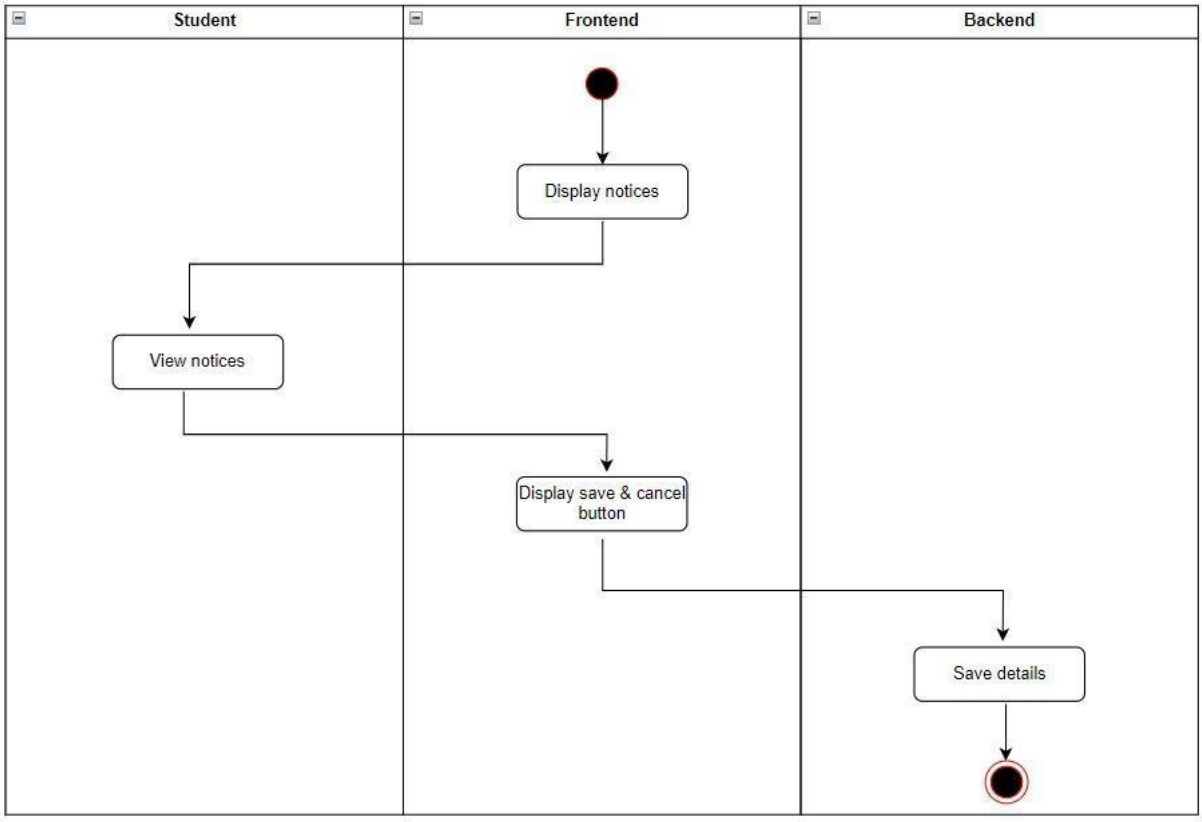


Figure 11: Activity diagram for Student to view notices

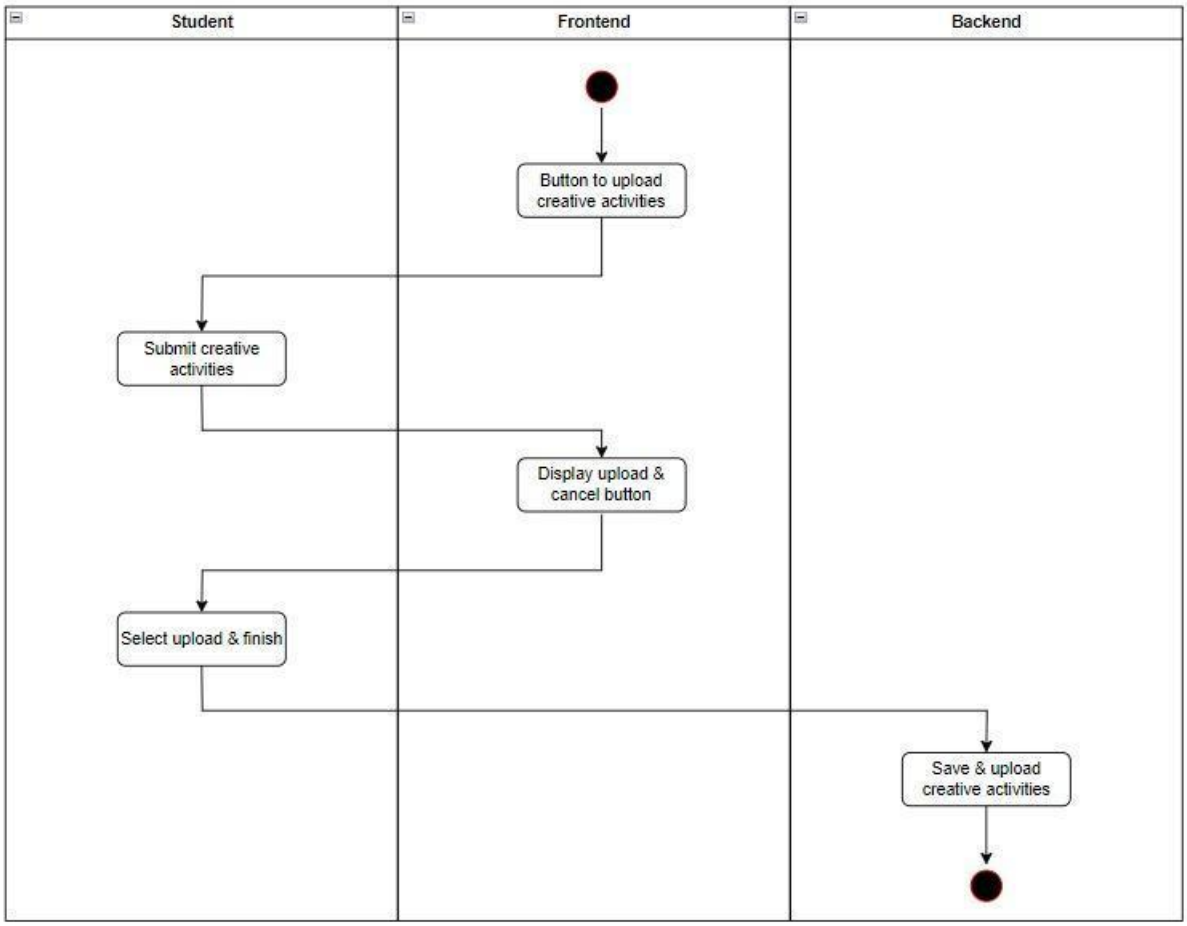


Figure 12: Activity diagram for Student to post creative activities

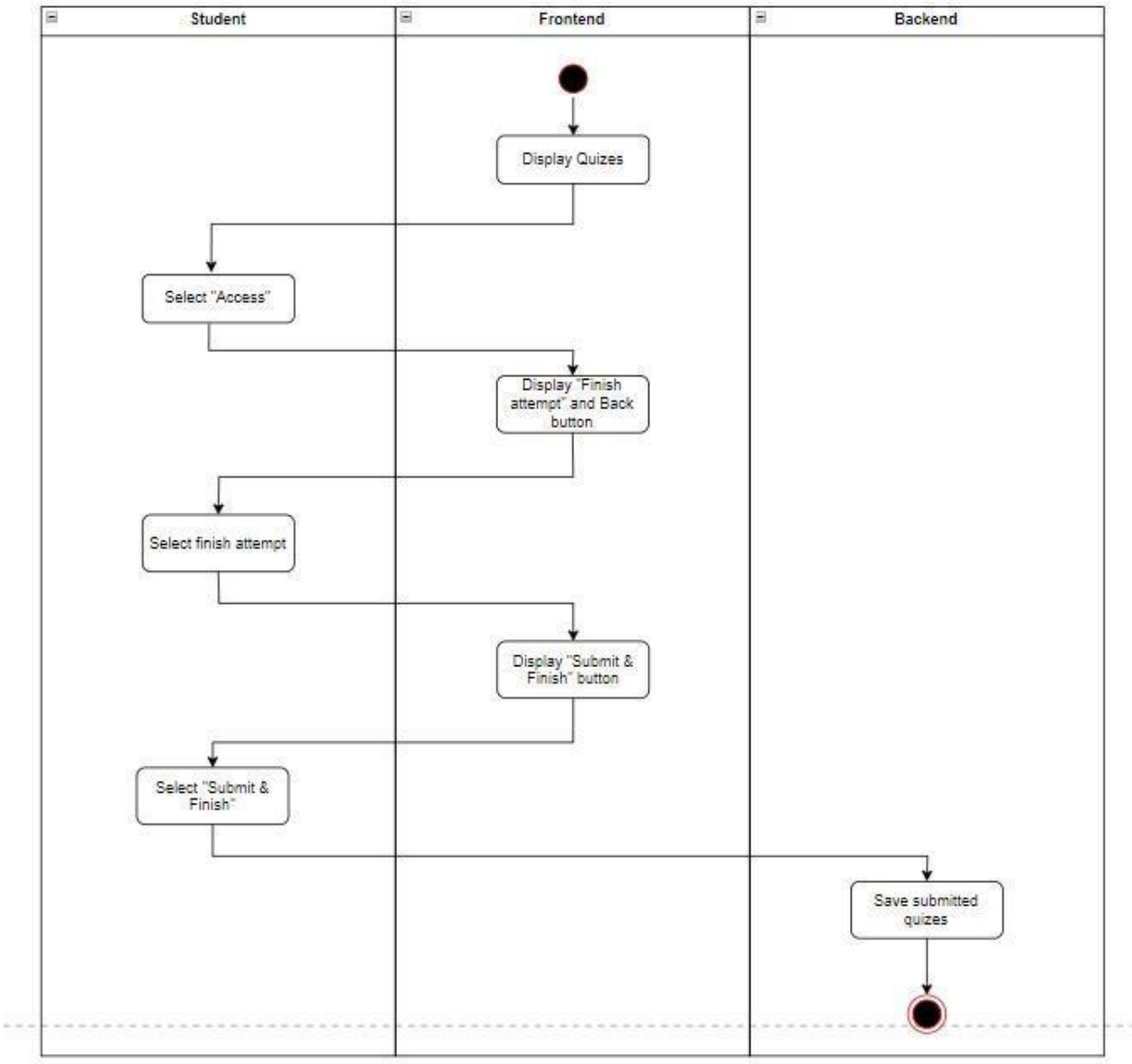


Figure 13: Activity diagram for Student to access to quizzes

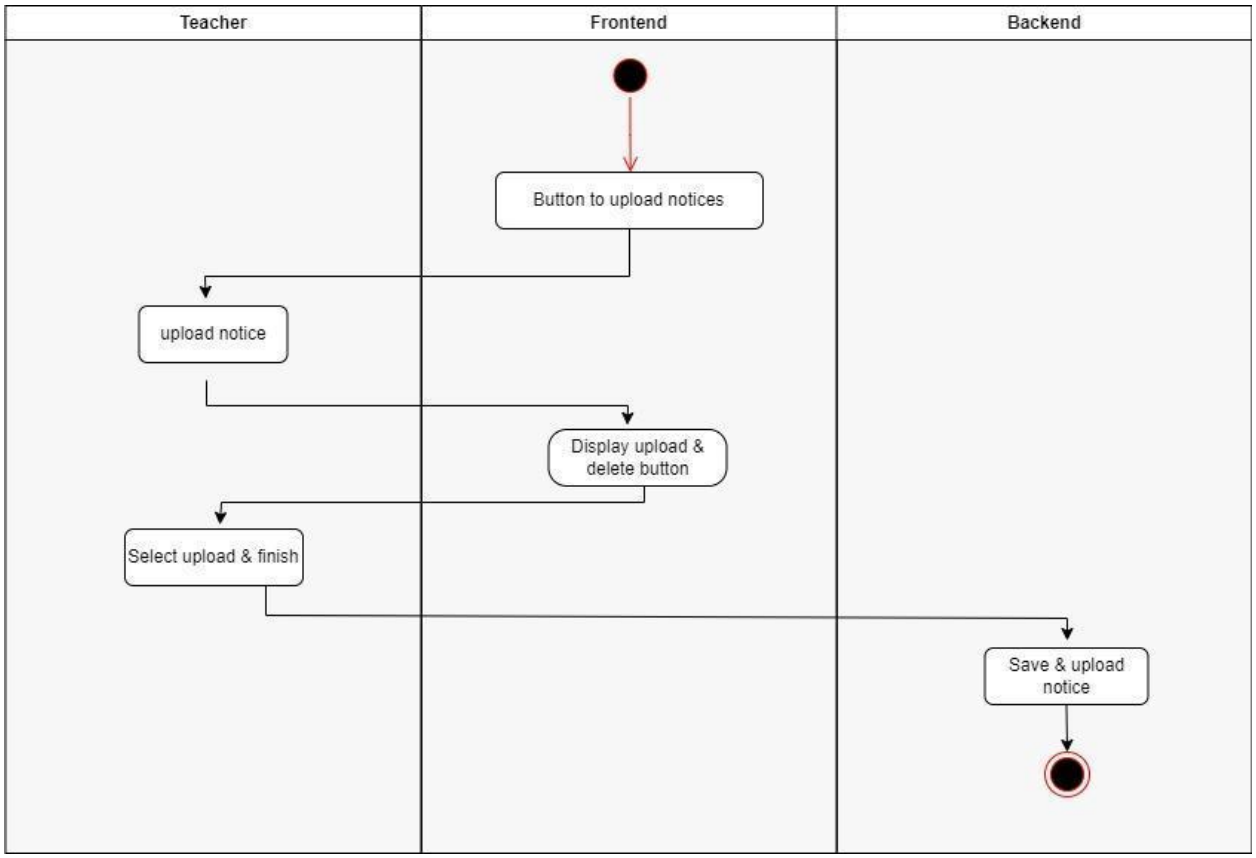


Figure 14: Activity diagram for Teacher to upload notices

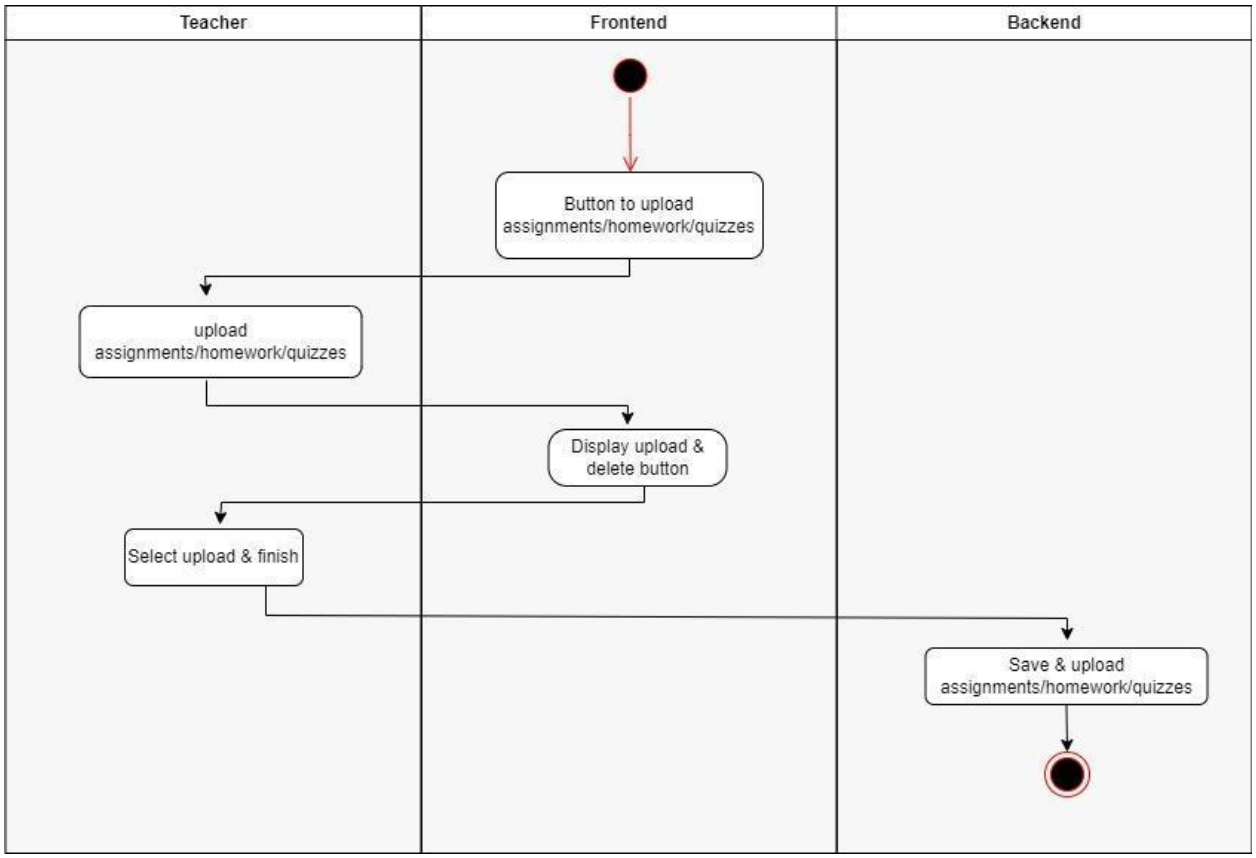


Figure 15: Activity diagram for Teacher to upload assignments/homework/quizzes

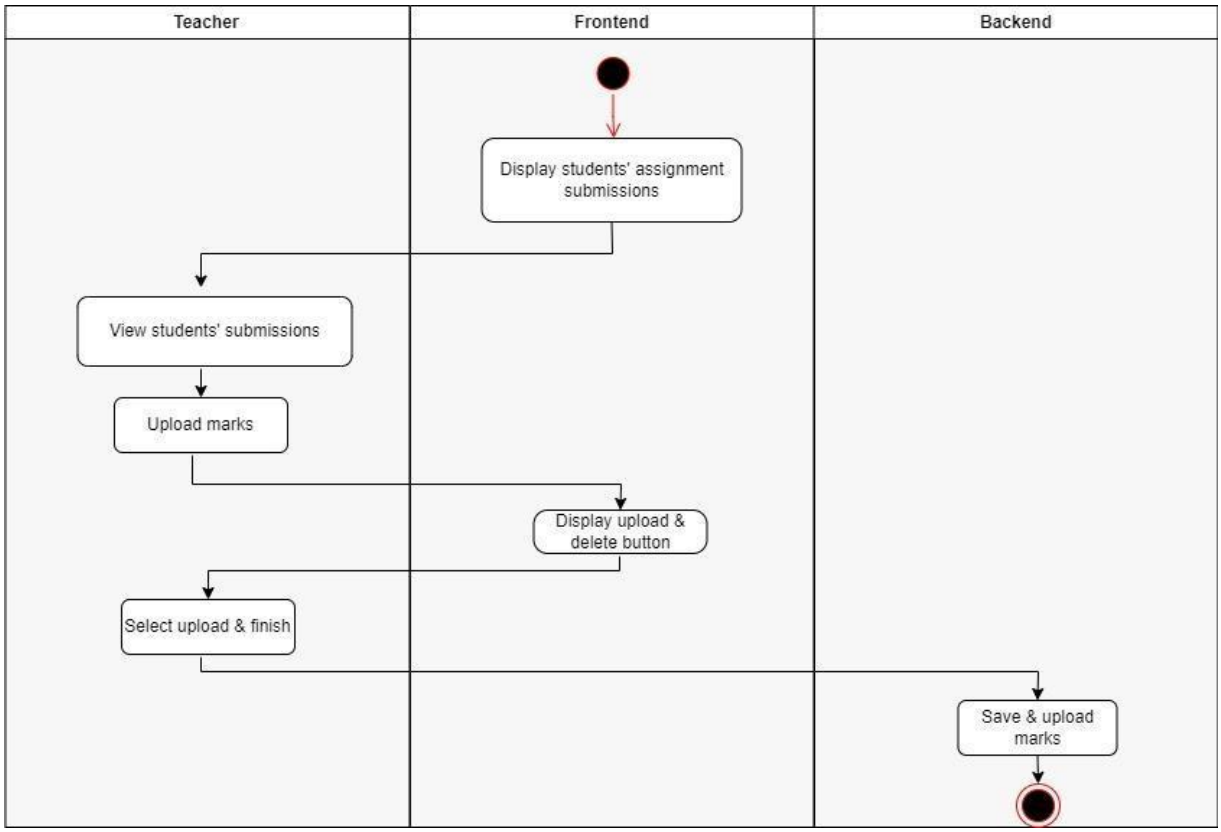


Figure 16: Activity diagram for Teacher to view submission of students

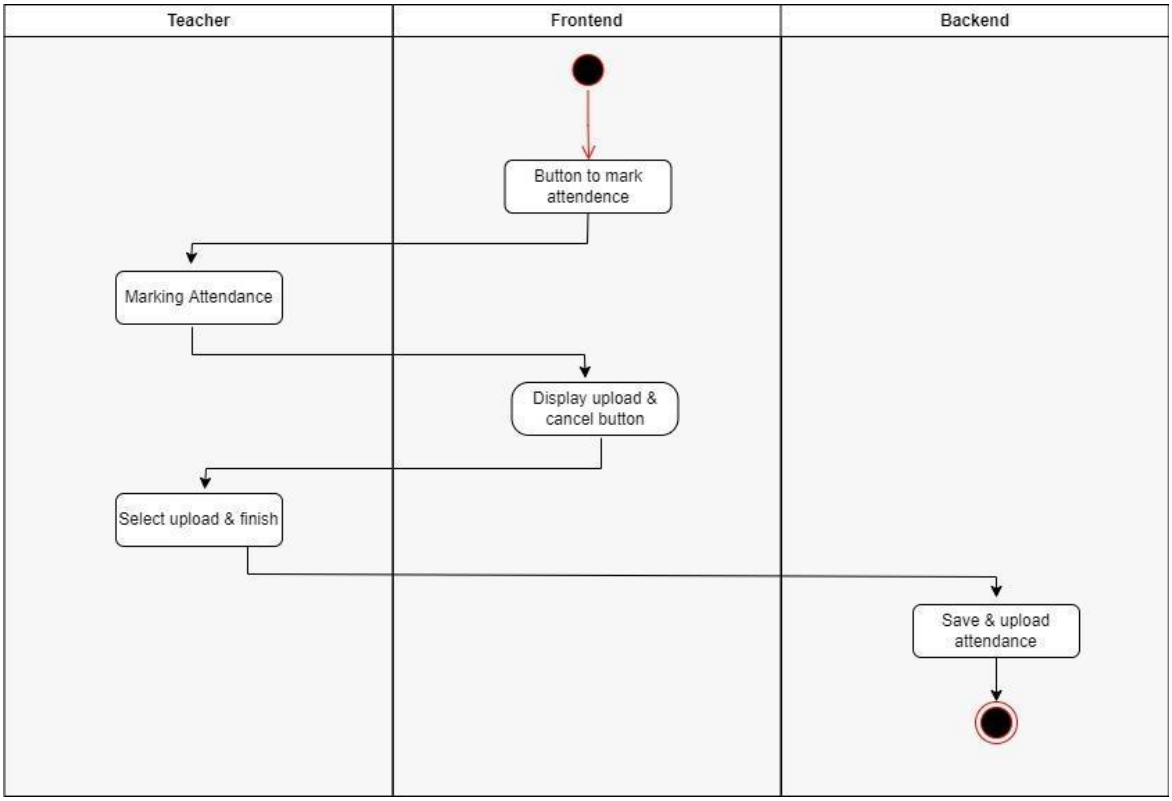


Figure 17: Activity diagram for Teacher to mark attendance

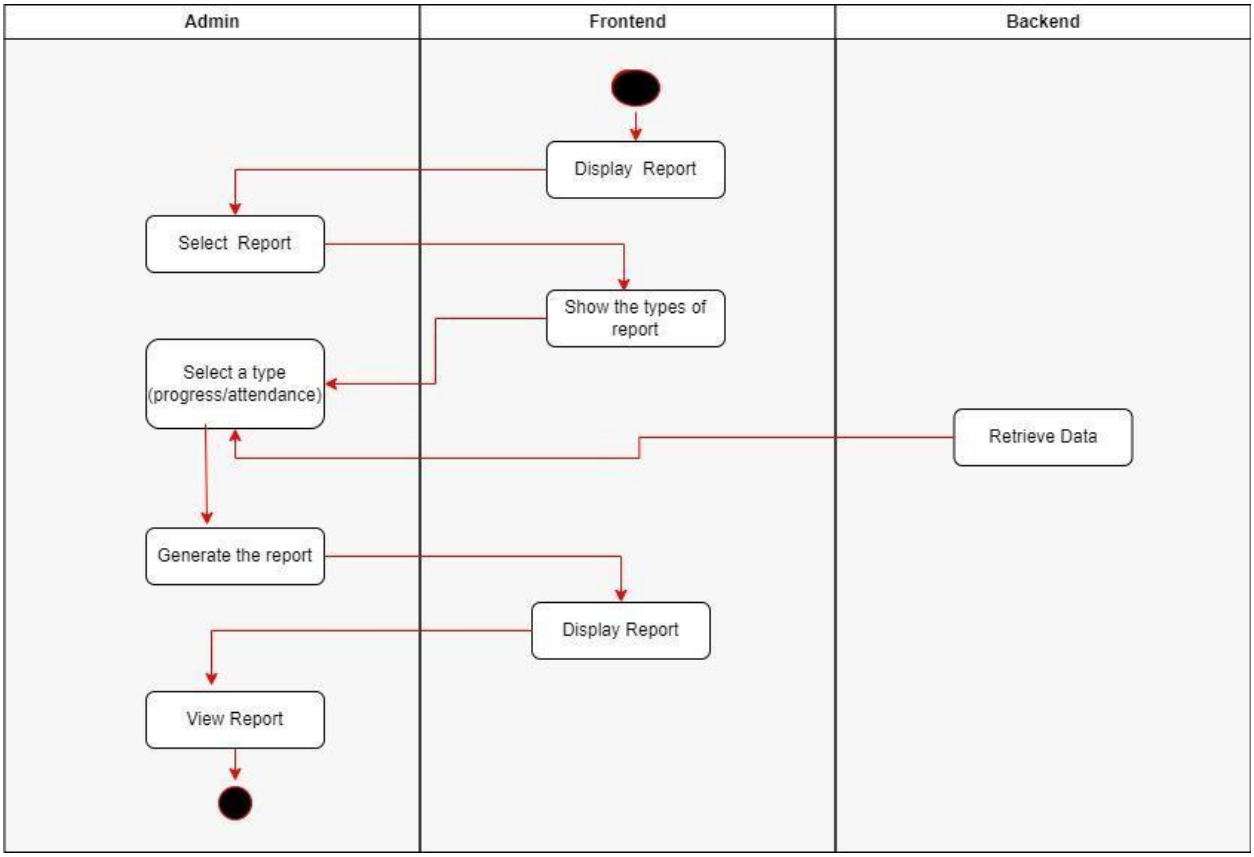


Figure 18: Activity diagram for Admin to generate and view reports

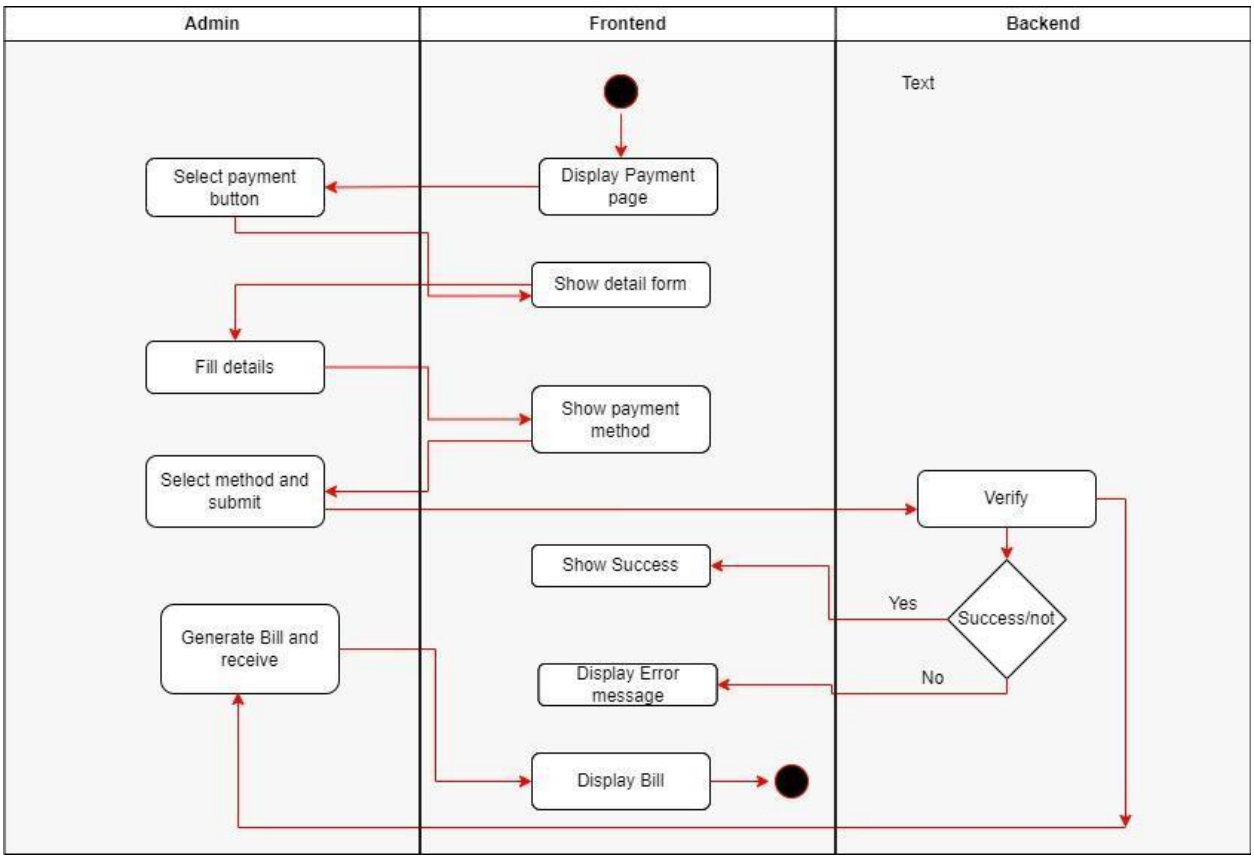


Figure 19: Activity diagram for Admin to handle payments

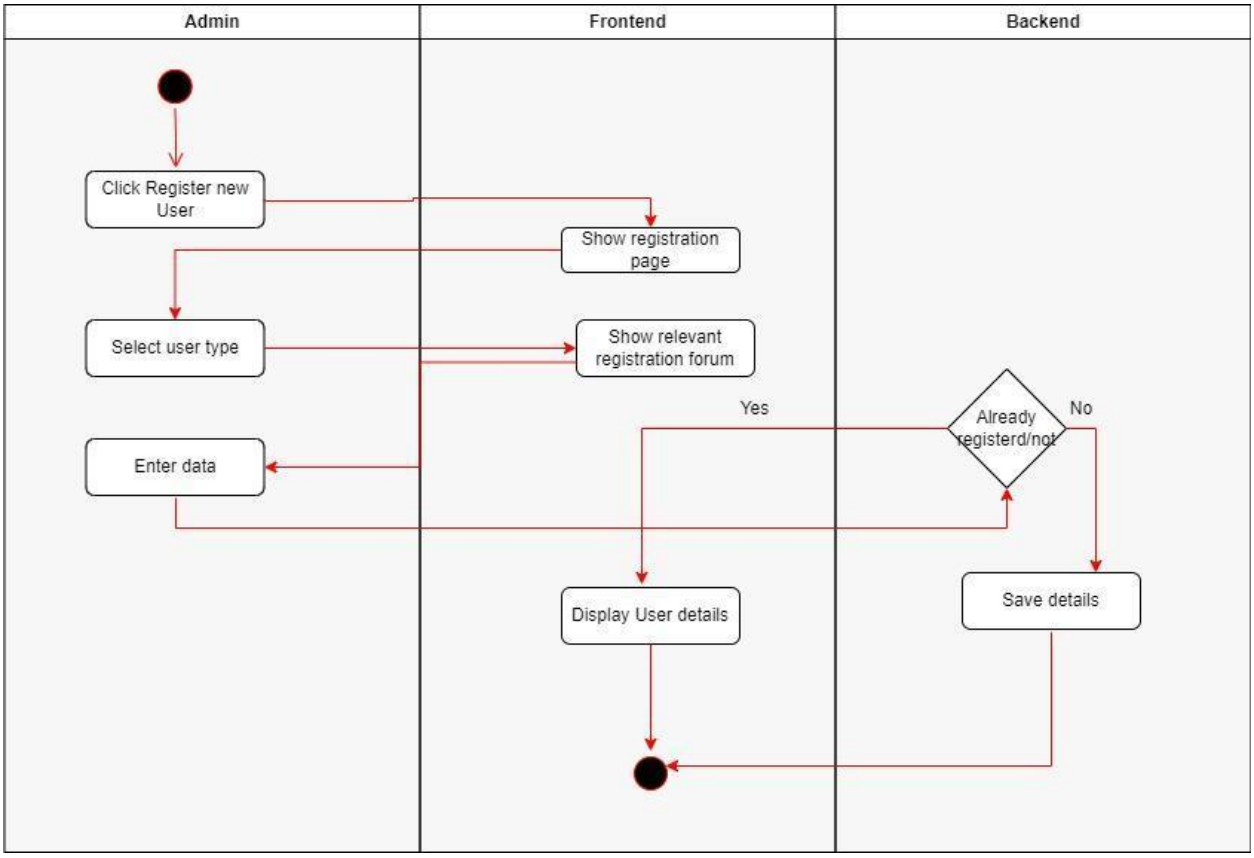


Figure 20: Activity diagram for Admin to handle user registration

5.3.5 Class diagram

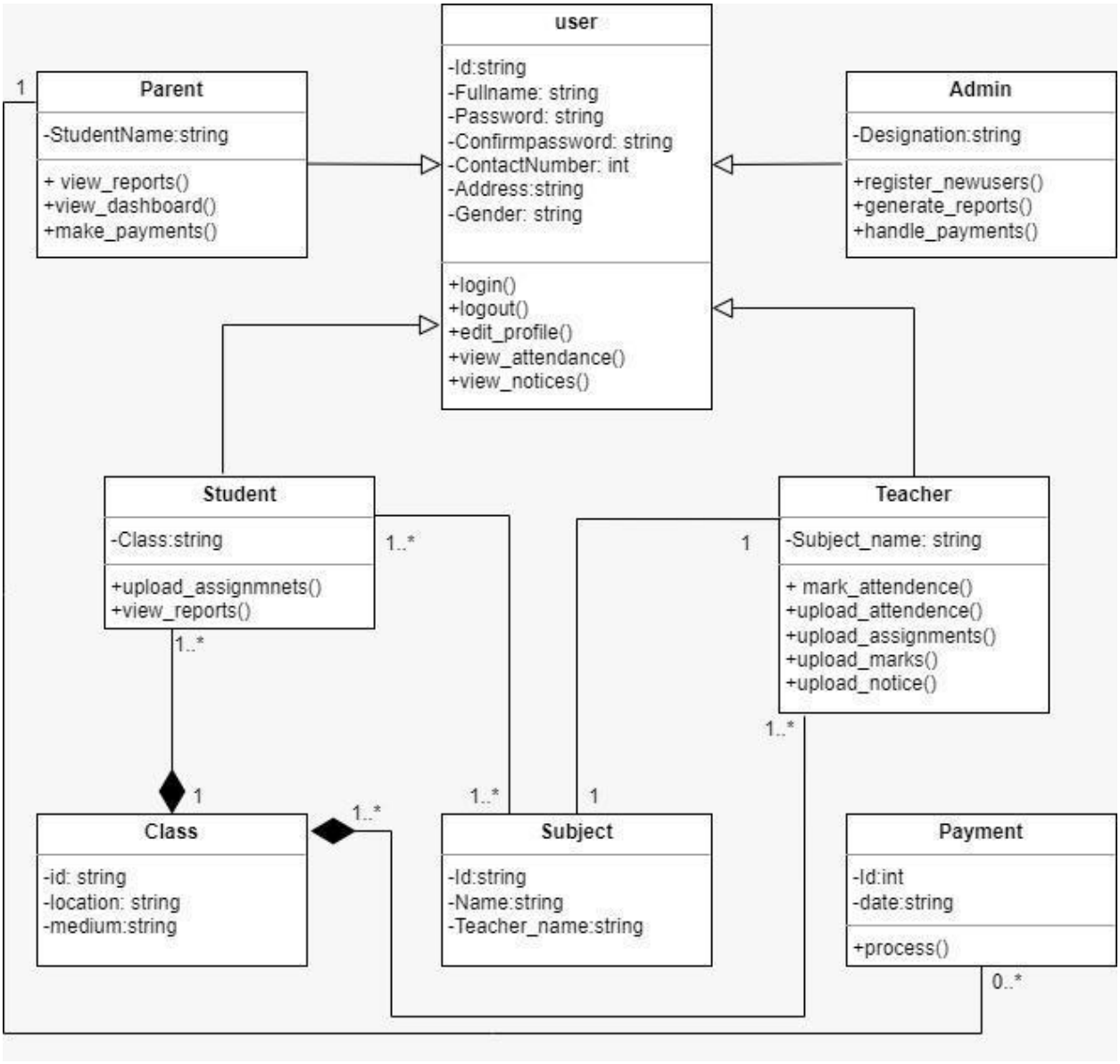


Figure 21: Class diagram

5.3.6 Sequence diagram

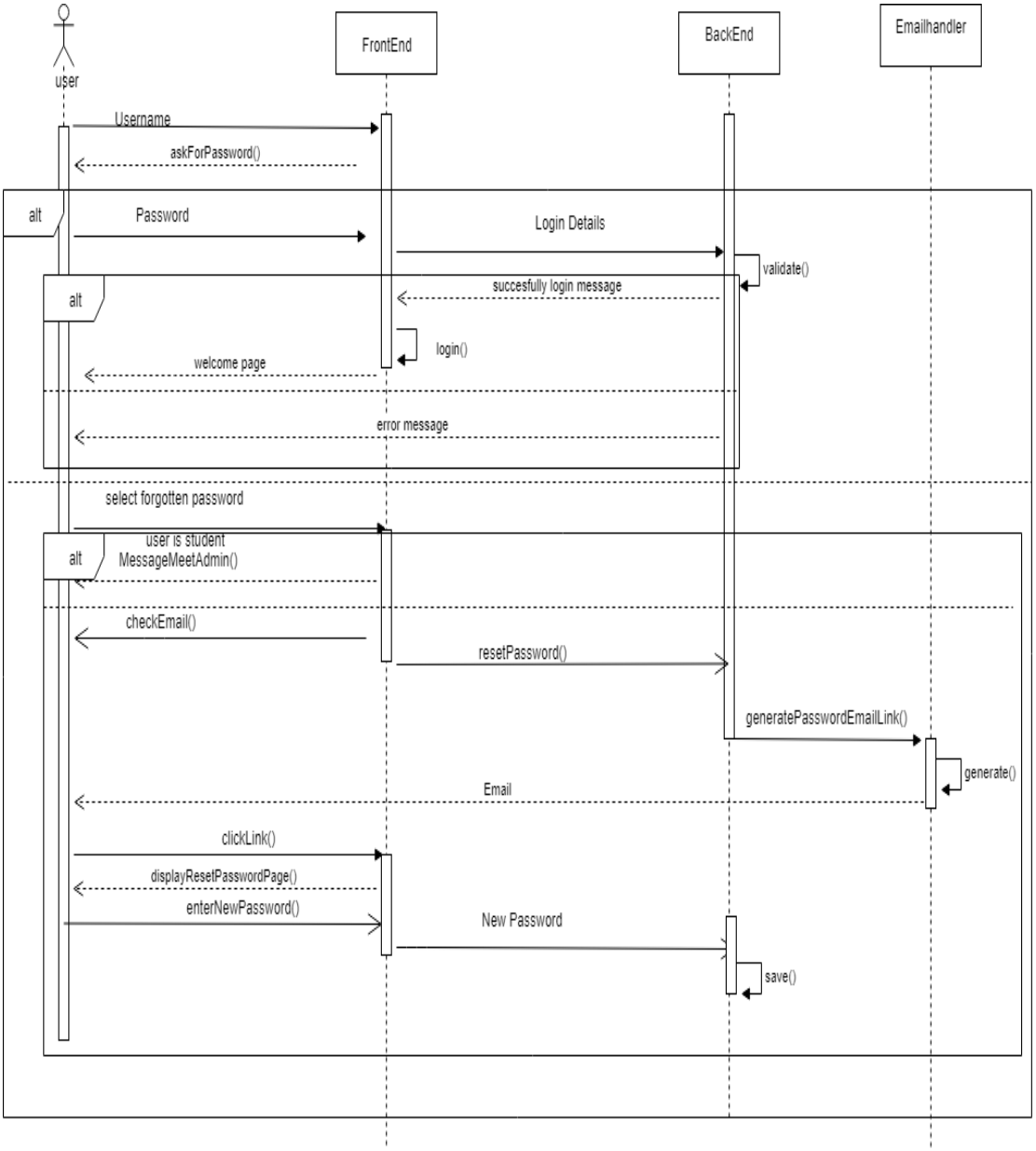


Figure 22: Login and reset password

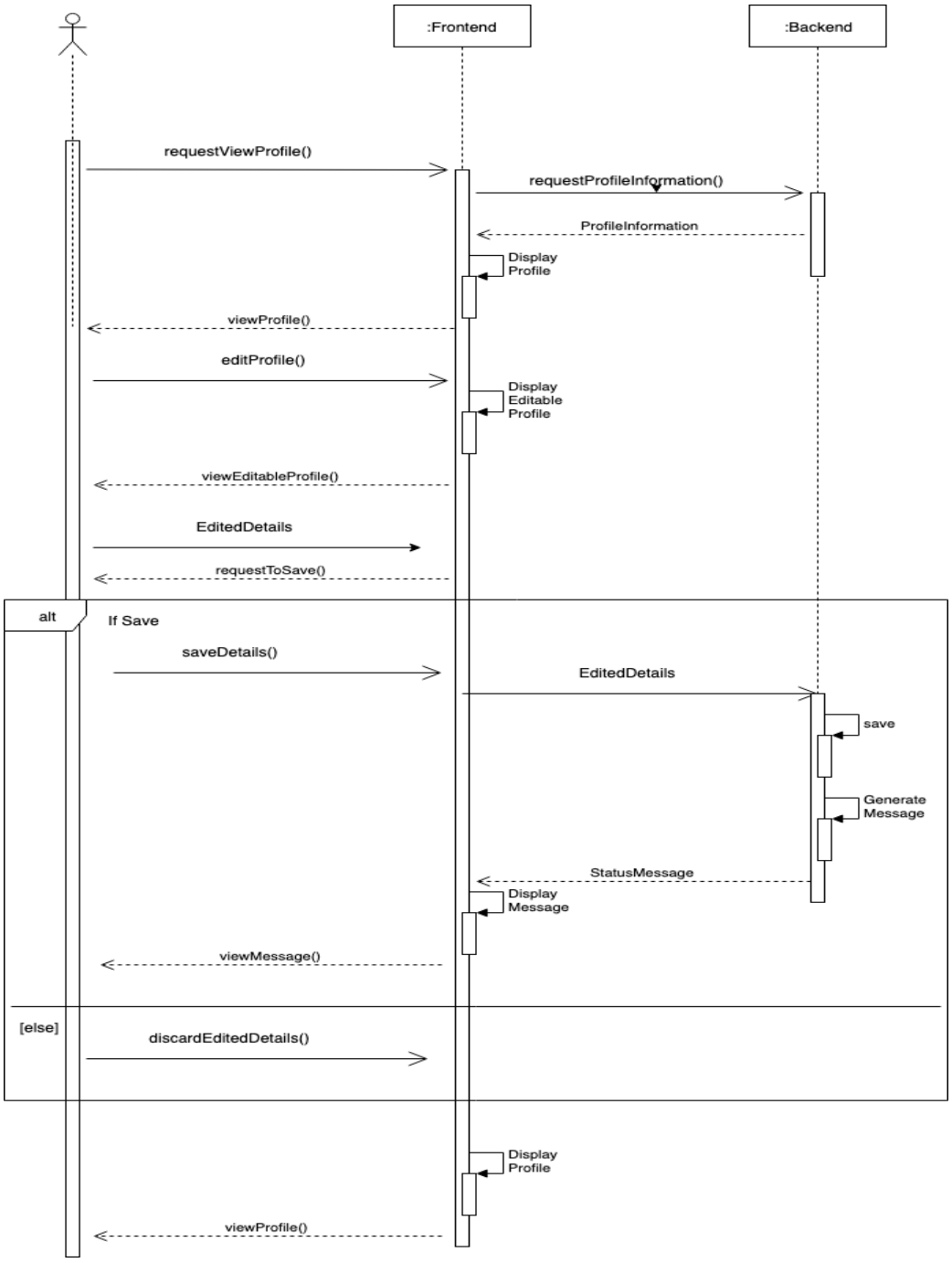


Figure 23: Edit user profile

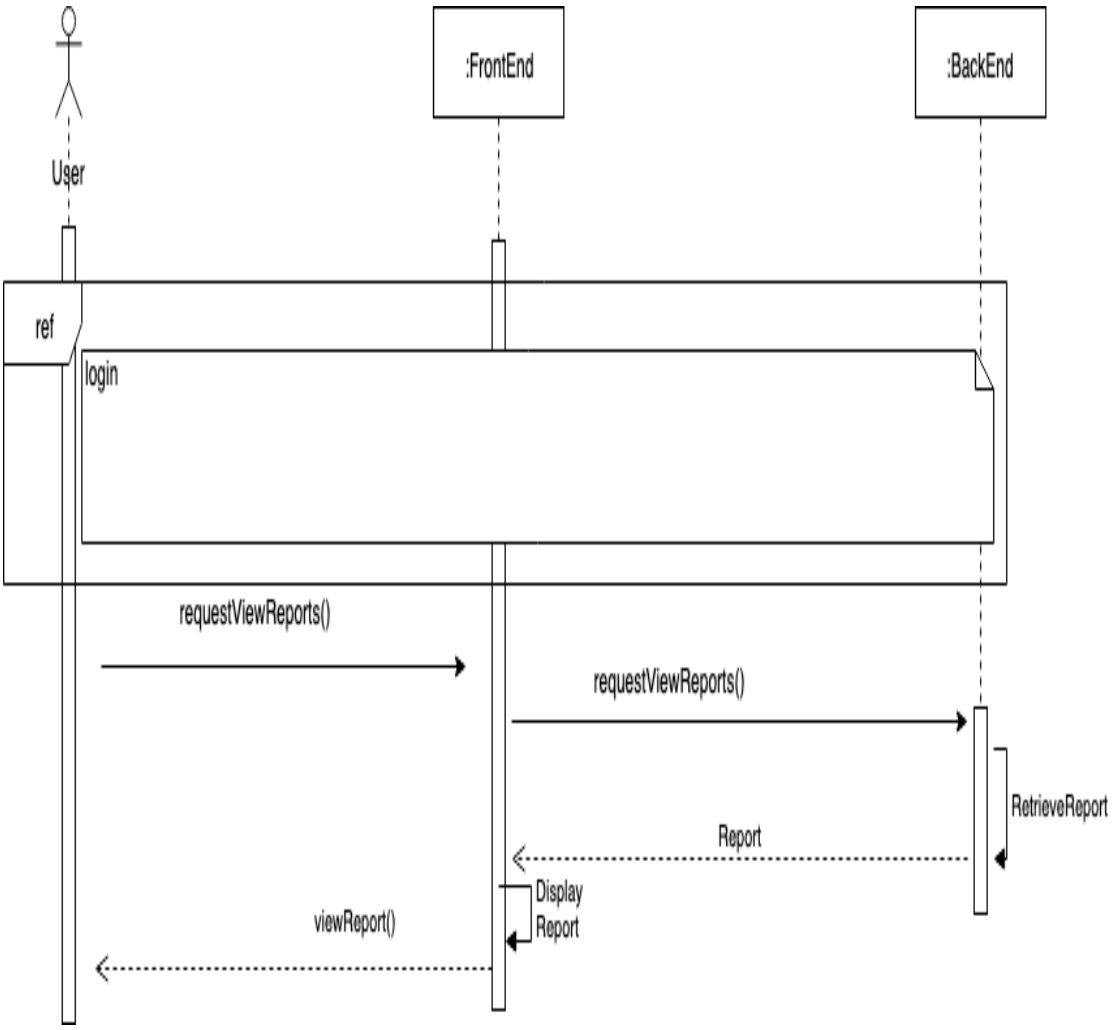


Figure 24: View reports

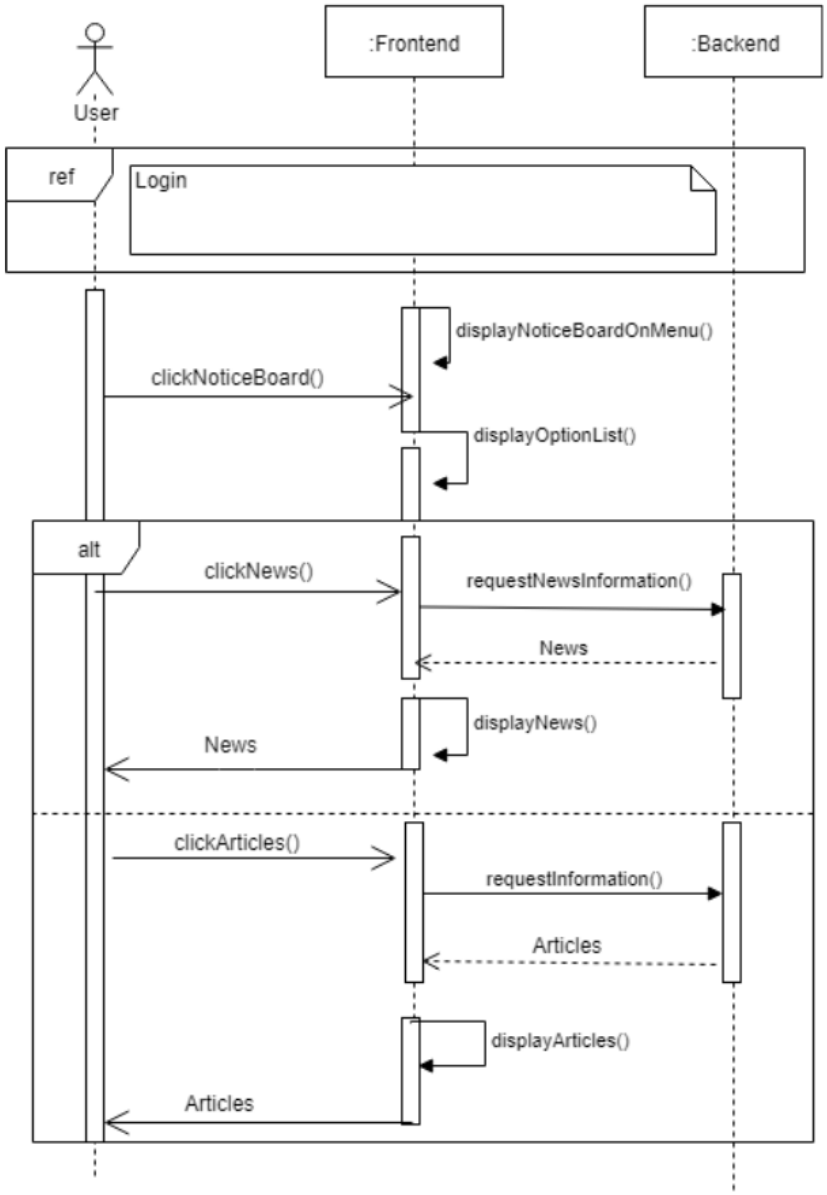


Figure 25: View articles and notices

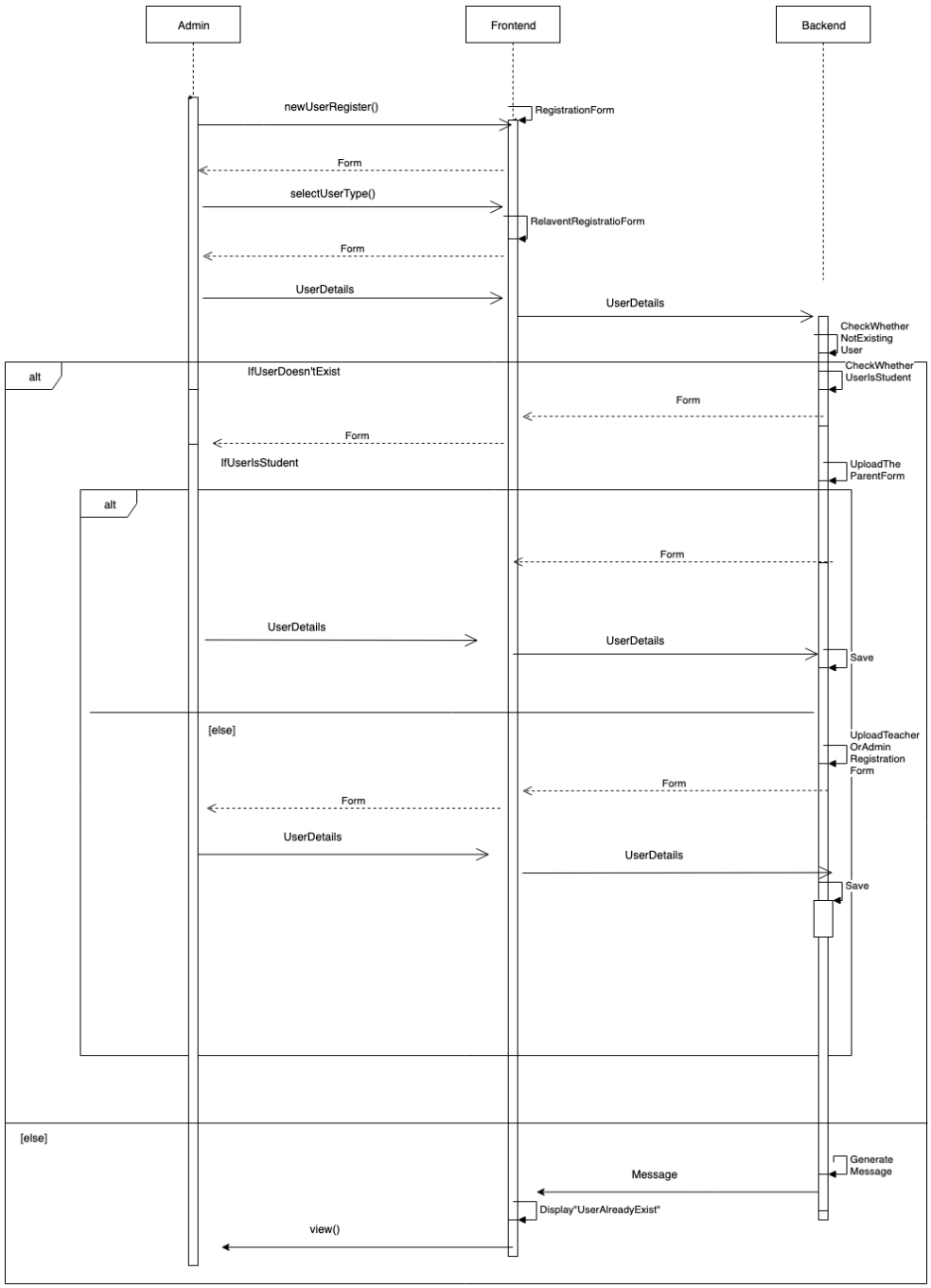


Figure 26: Register new user

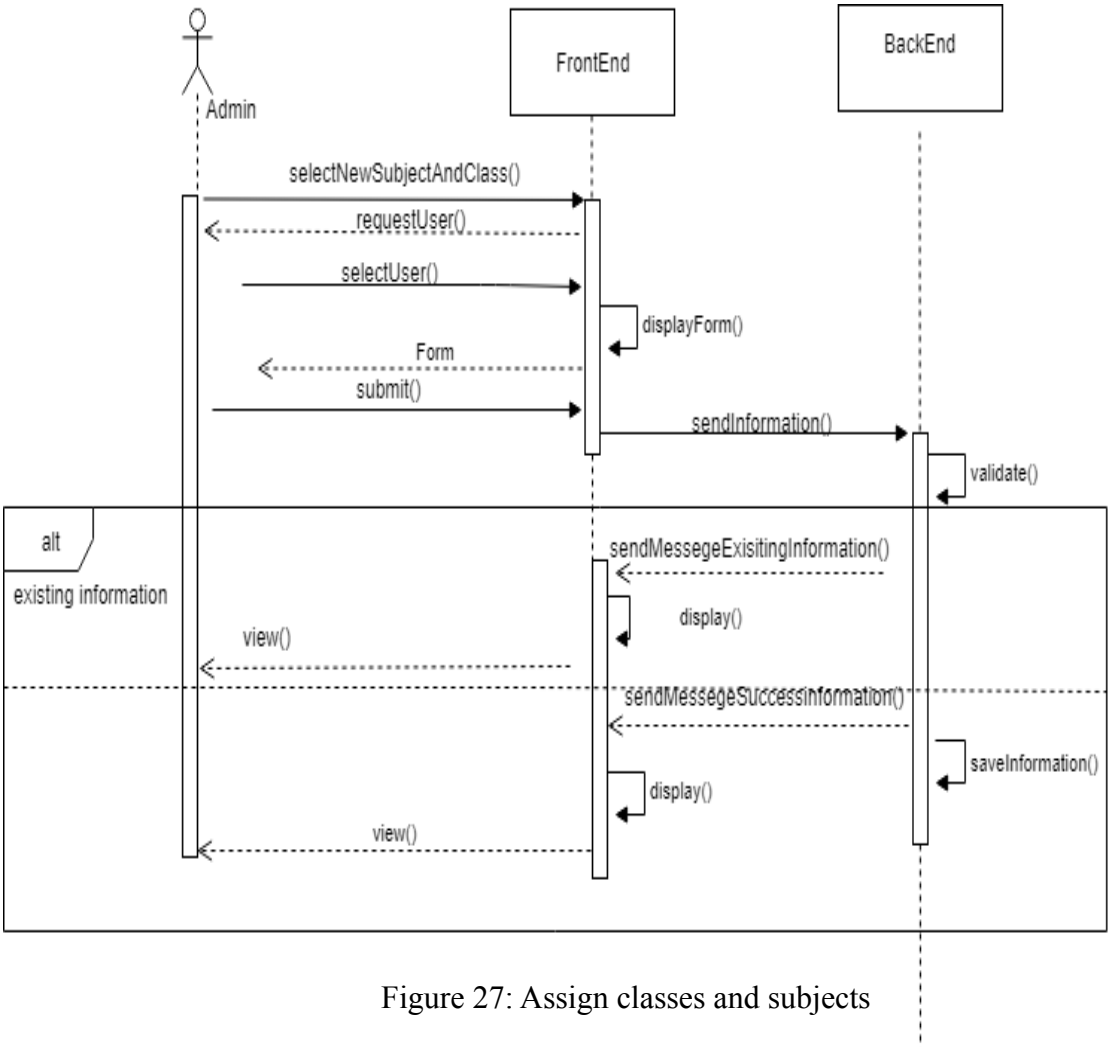


Figure 27: Assign classes and subjects

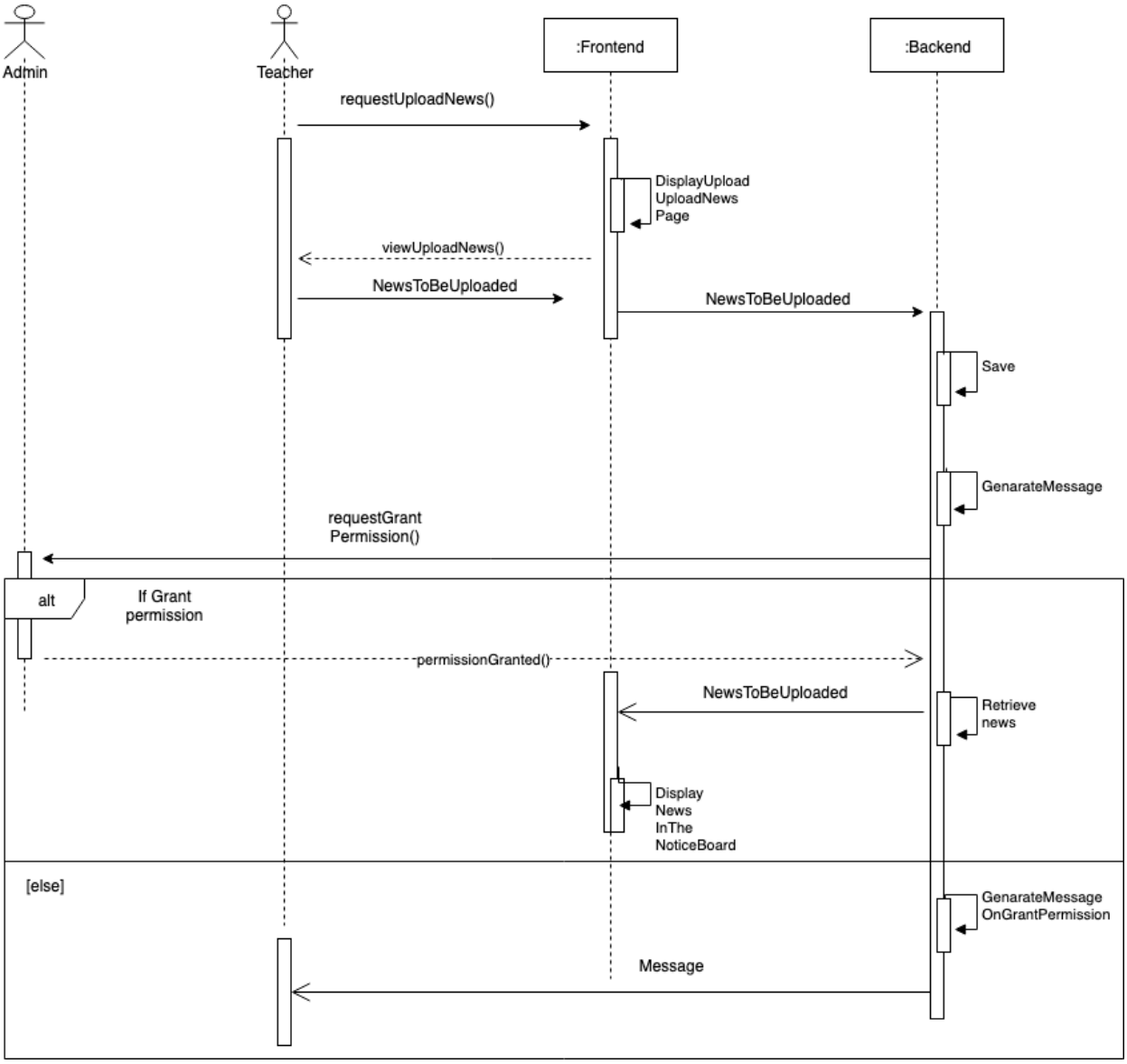


Figure 28: Upload activities

self call

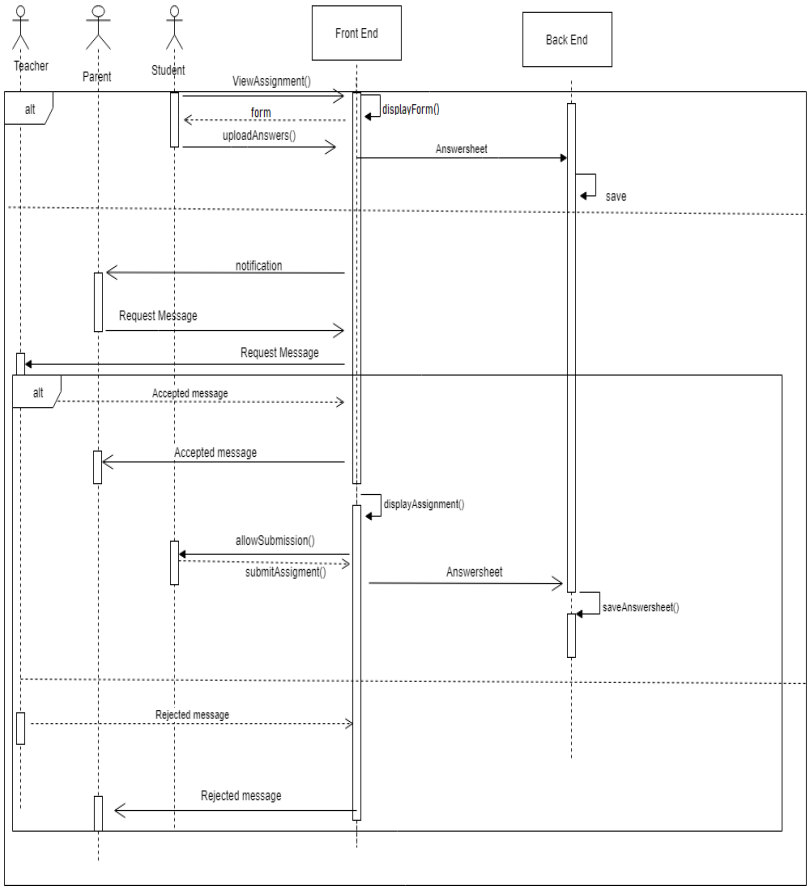


Figure 29: View assignment and upload answers

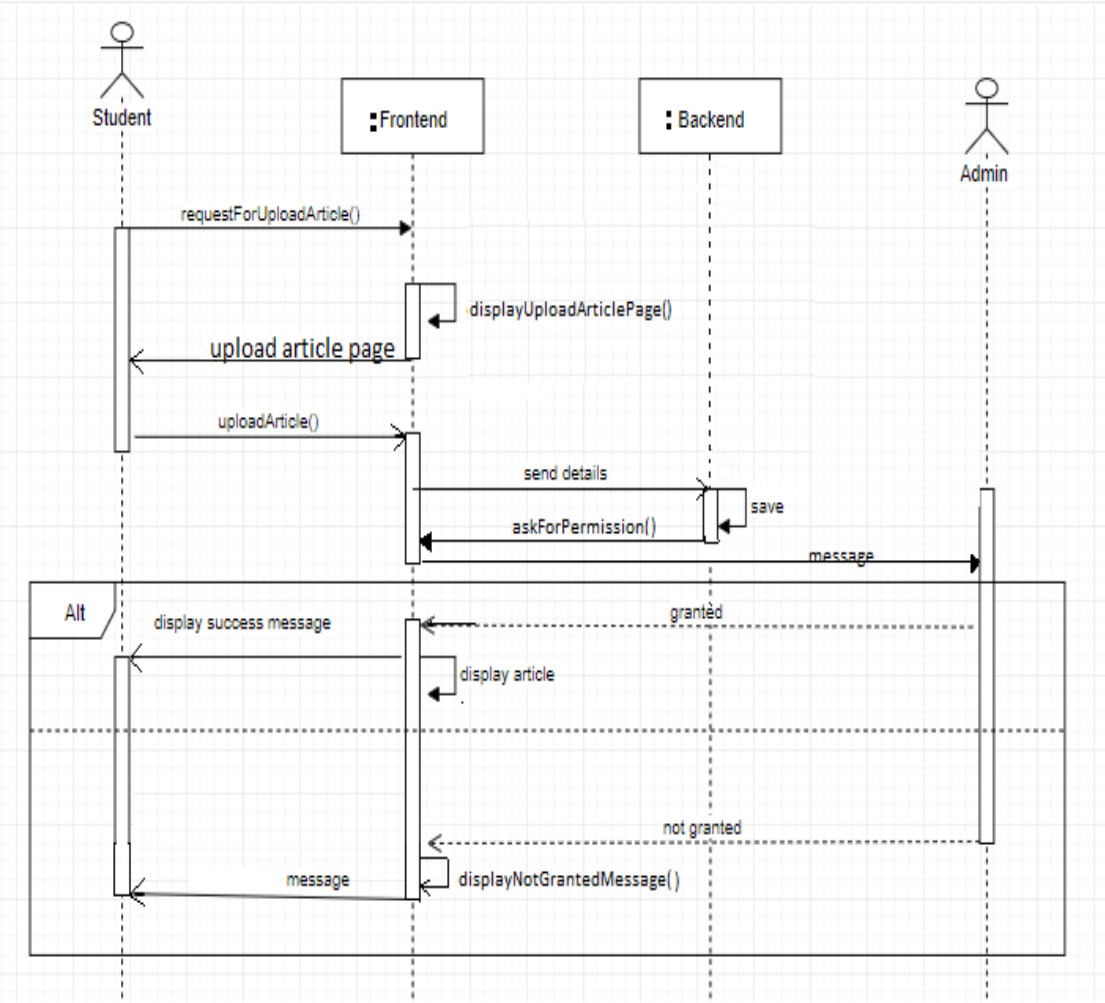


Figure 30: Upload articles

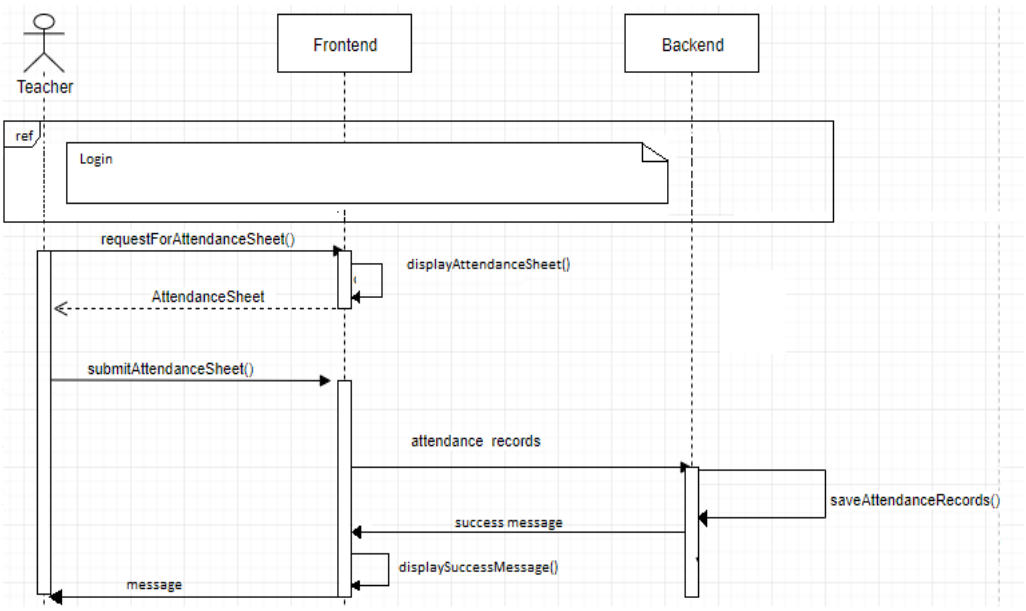


Figure 32: Upload learning materials

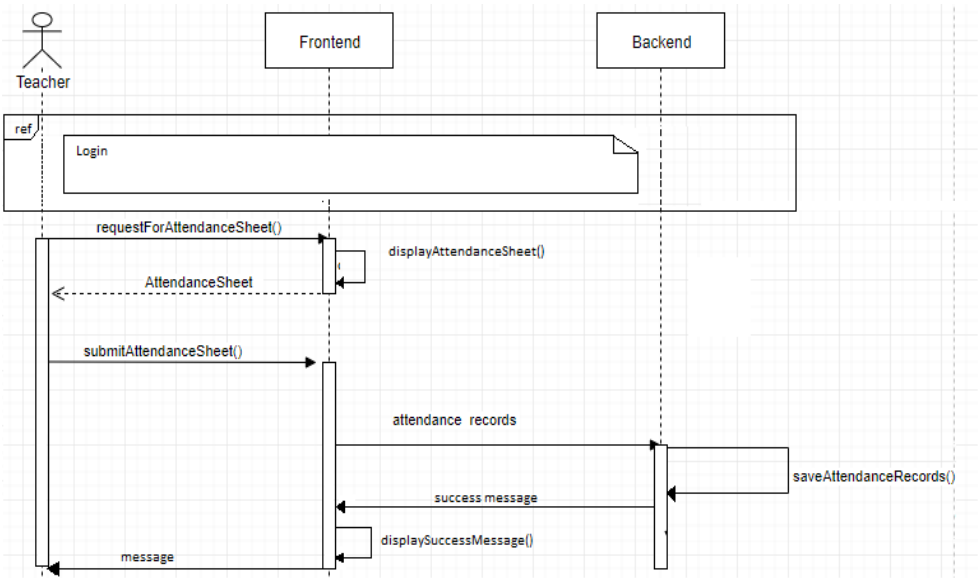


Figure 33: Upload news

5.4 Summary

In the UML diagrams and the EER diagram it shows design details of our system. In the EER diagram it shows what the entities in the database are and how they are related. Use case diagram shows all the requirements from user's point of view. Activity diagram shows flow of activities in the system. Class diagram shows the classes and their relationships. Sequence diagram shows timebased dynamics of interactions

Chapter 5

Implementation

5.1 Introduction

After identifying user requirements, we had a meeting with our client and we modified some user requirements. We selected the agile scrum as our software process model. First, we selected the technologies that we can use to fulfil the user requirements and had a knowledge about those technologies. Then we drew the UML diagrams and EER diagram in order to analyze user requirements. Also, we have drew use case diagram, class diagram, activity diagrams and sequence diagrams which are regarding to our system. Then we corrected those diagrams according to the comments given by our supervisor. Our system consists of two main modules as web application and mobile application. Currently we are designing the database and part of web application.

5.1.1 Interfaces

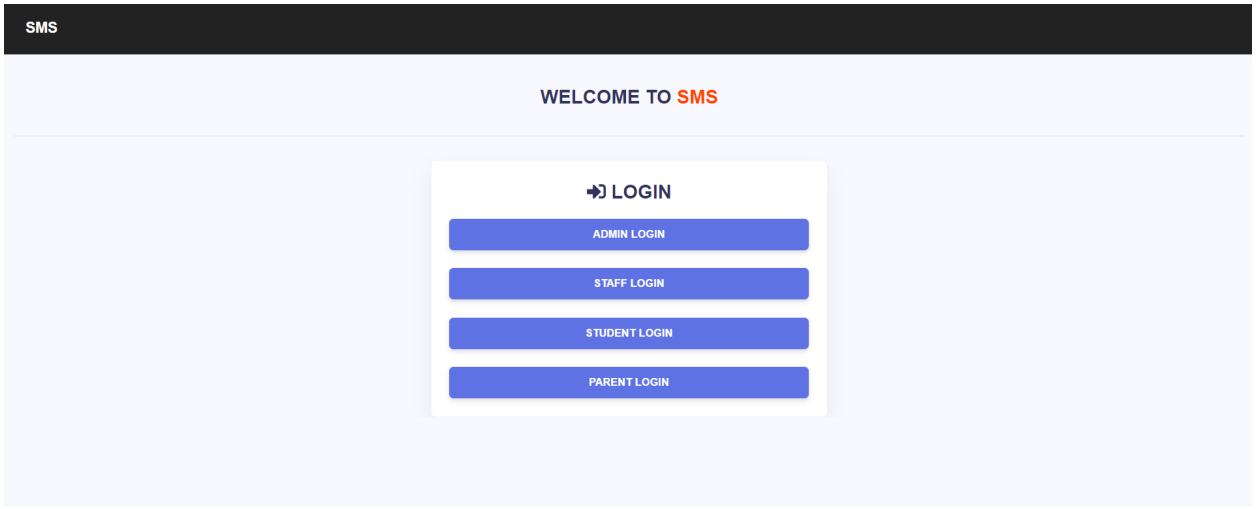


Figure 34: UI for user login

5.1.1. Admin Module

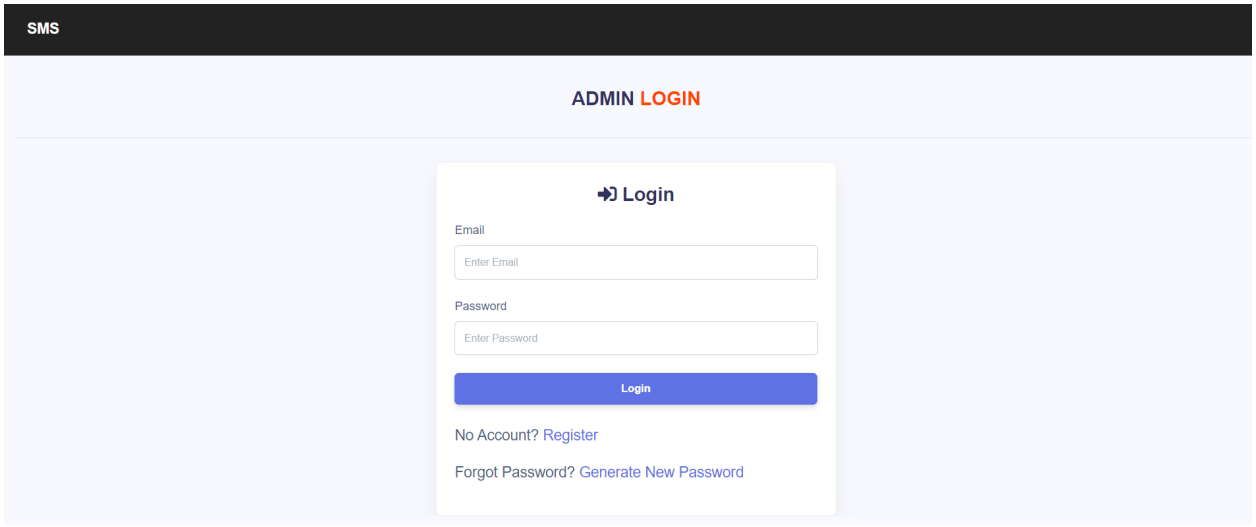


Figure 35: UI for Admin login

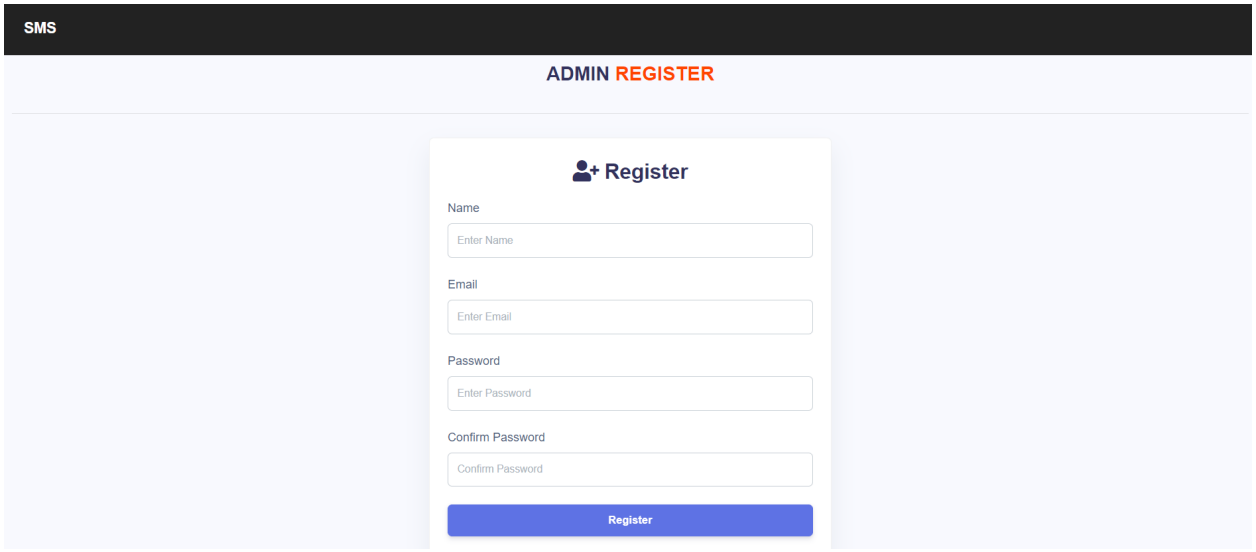


Figure 36: UI for Admin registration

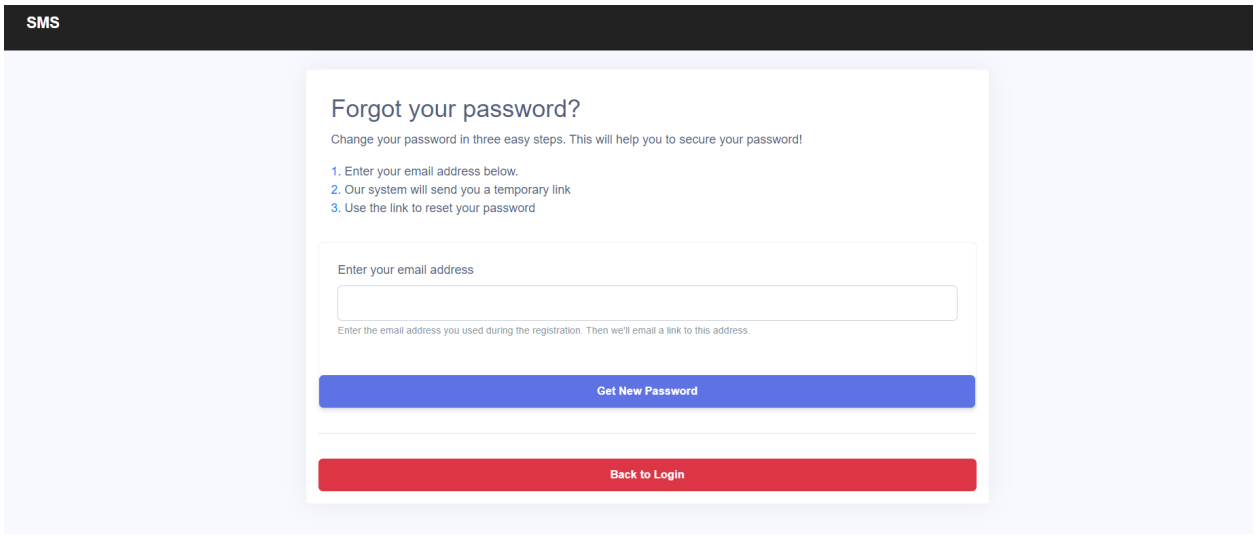


Figure 37: UI for Forgot password

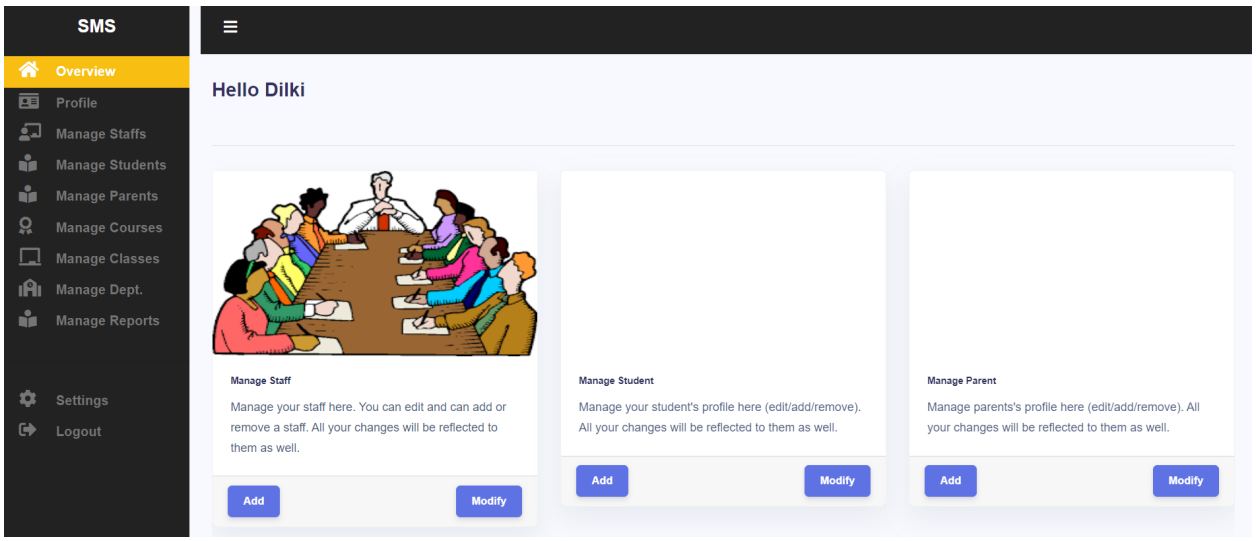


Figure 38: UI for Admin overview

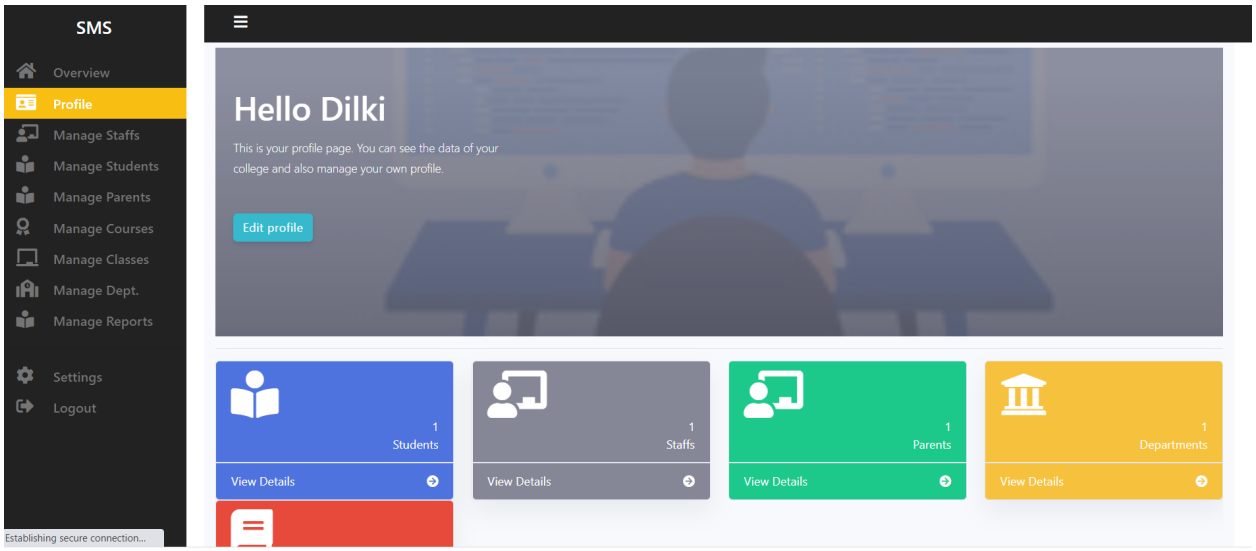


Figure 39: UI for Admin profile

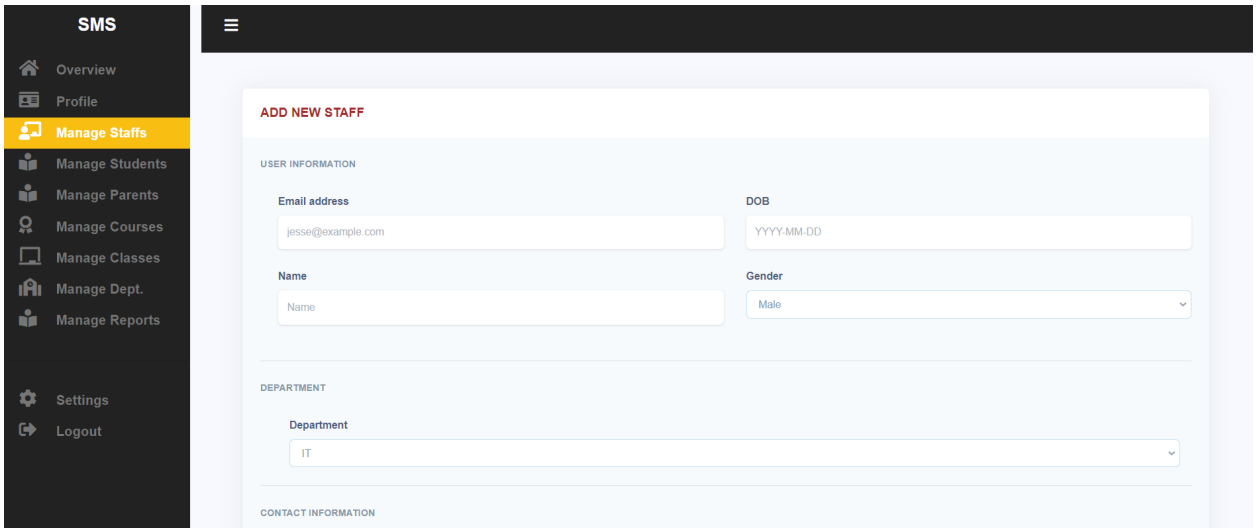


Figure 40: UI for manage profiles

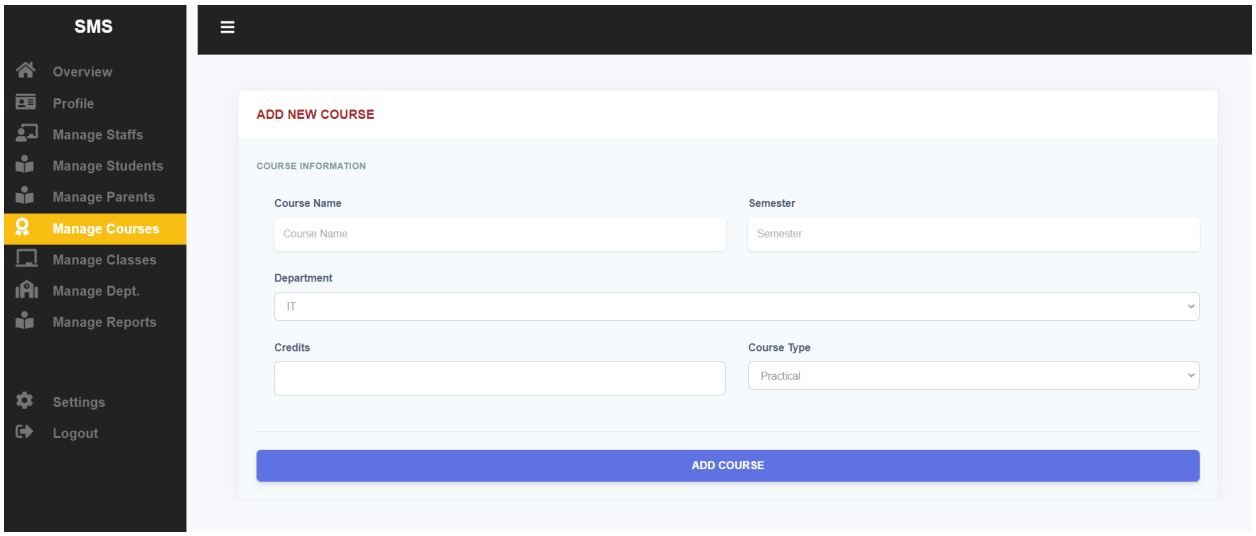


Figure 41: UI for manage courses

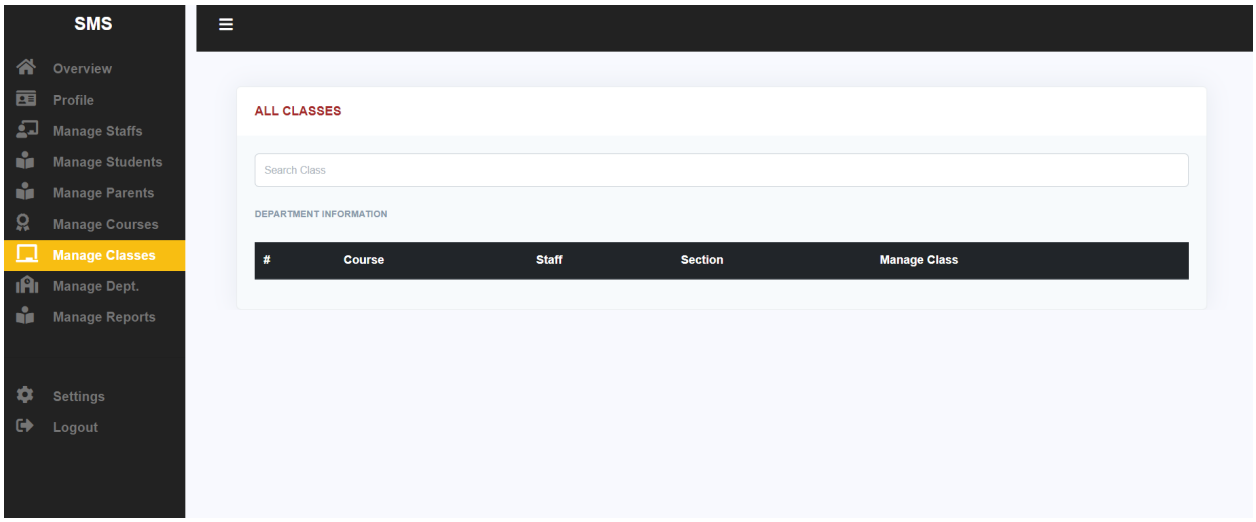


Figure 42: UI for manage classes

5.1.2. Student Module

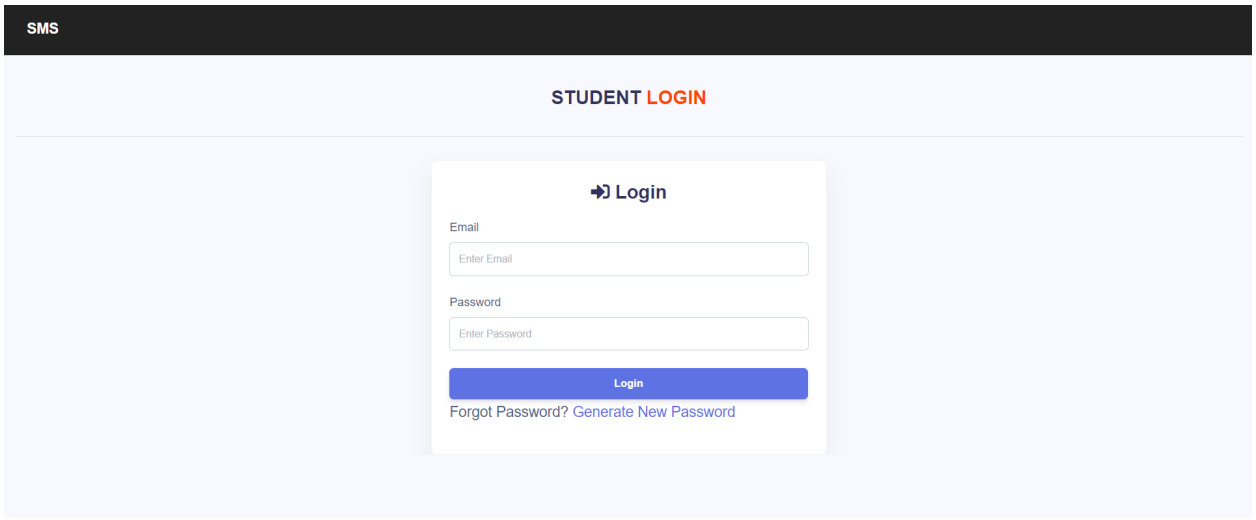


Figure 43: UI for student login

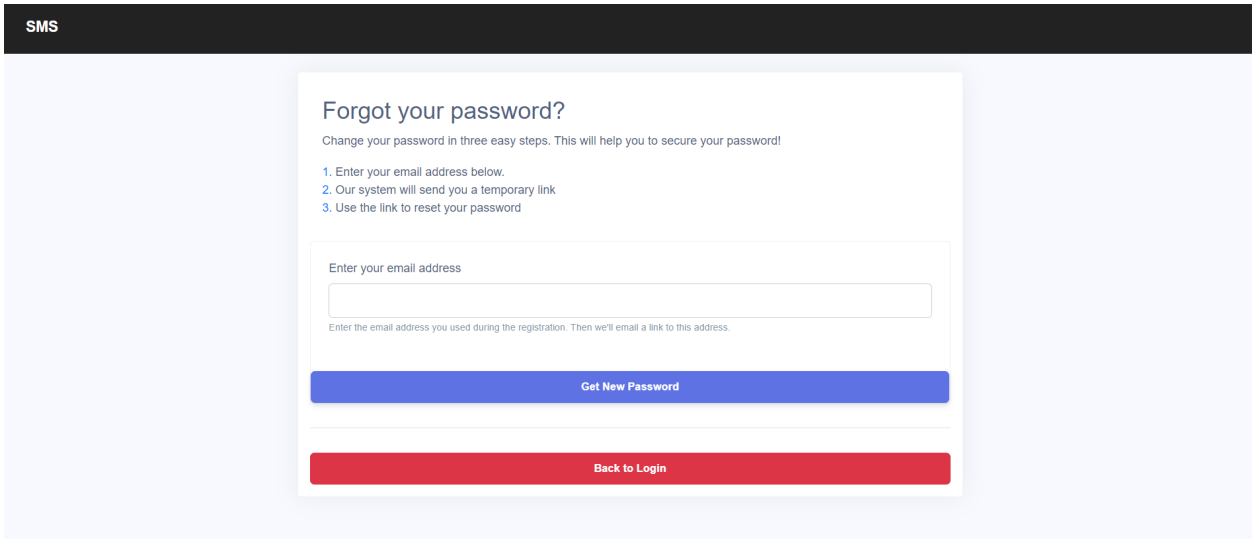


Figure 44: UI for student forgot password

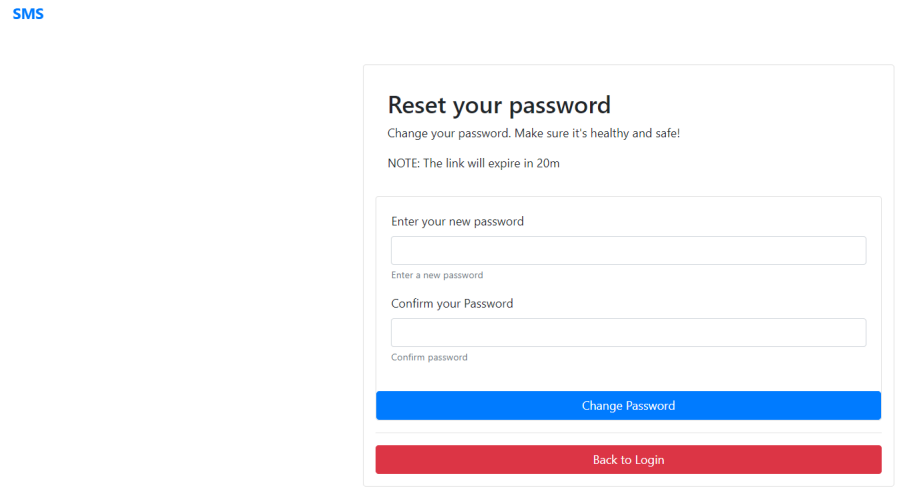


Figure 45: UI for student reset password

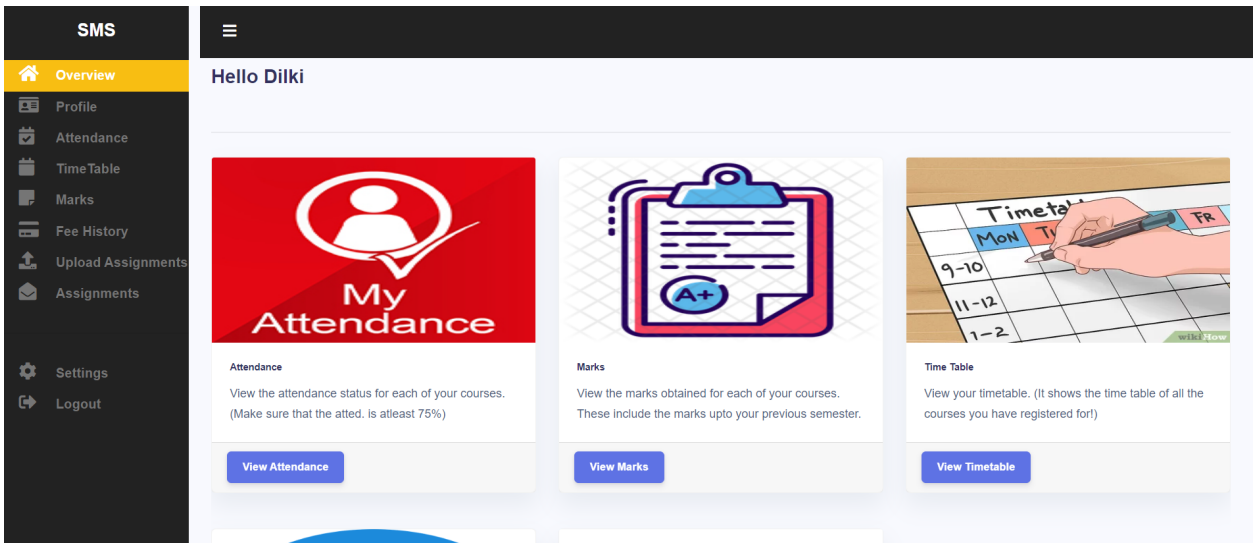


Figure 46: UI for student overview

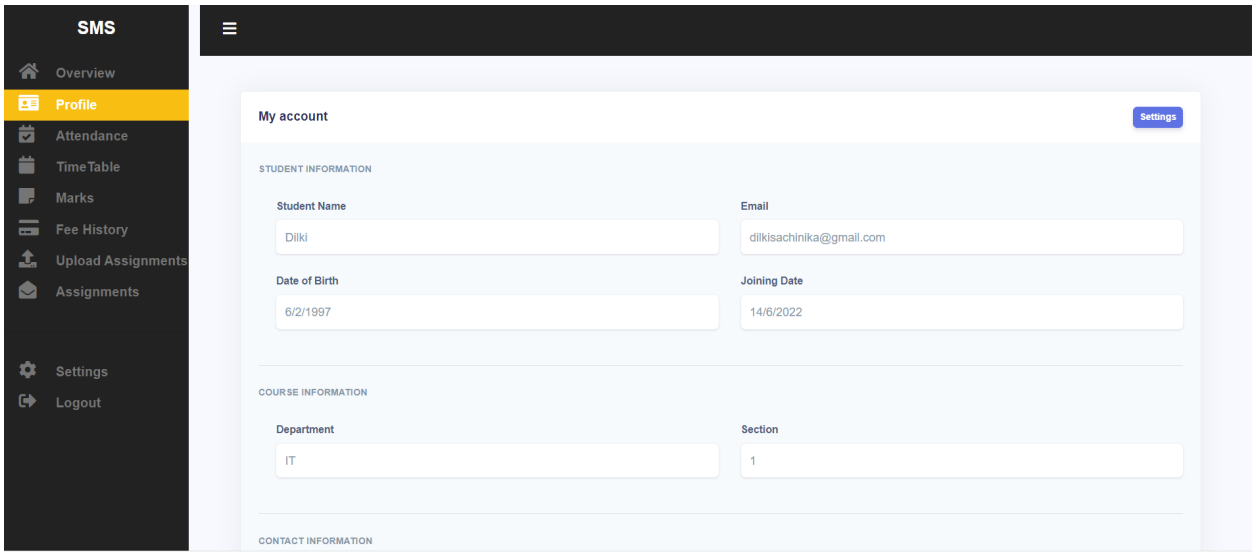


Figure 46: UI for student profile

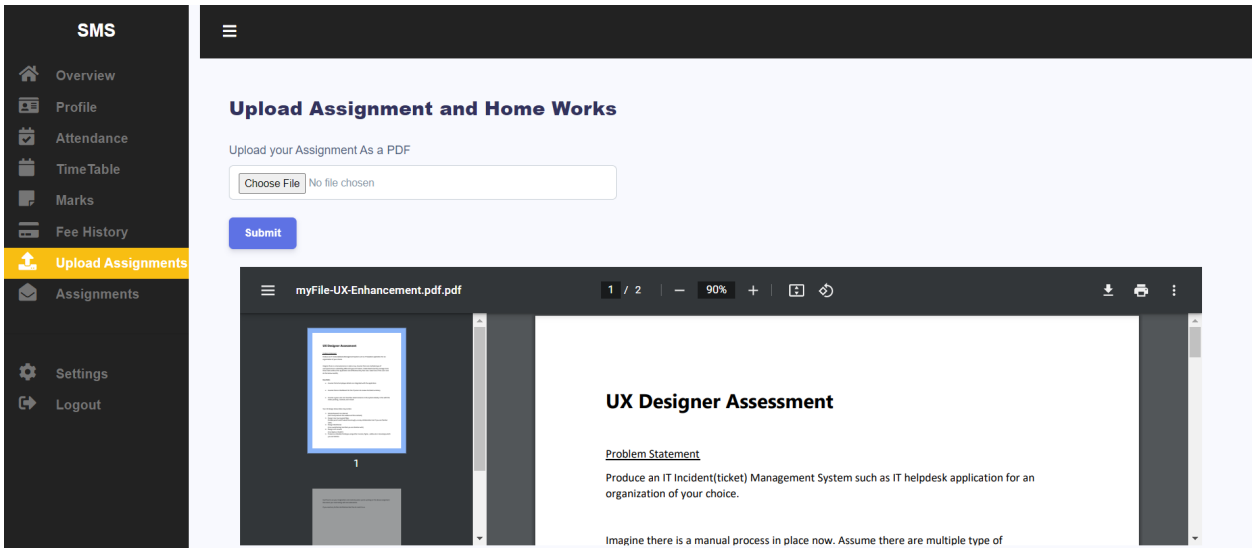


Figure 47: UI for student upload assignments and homework

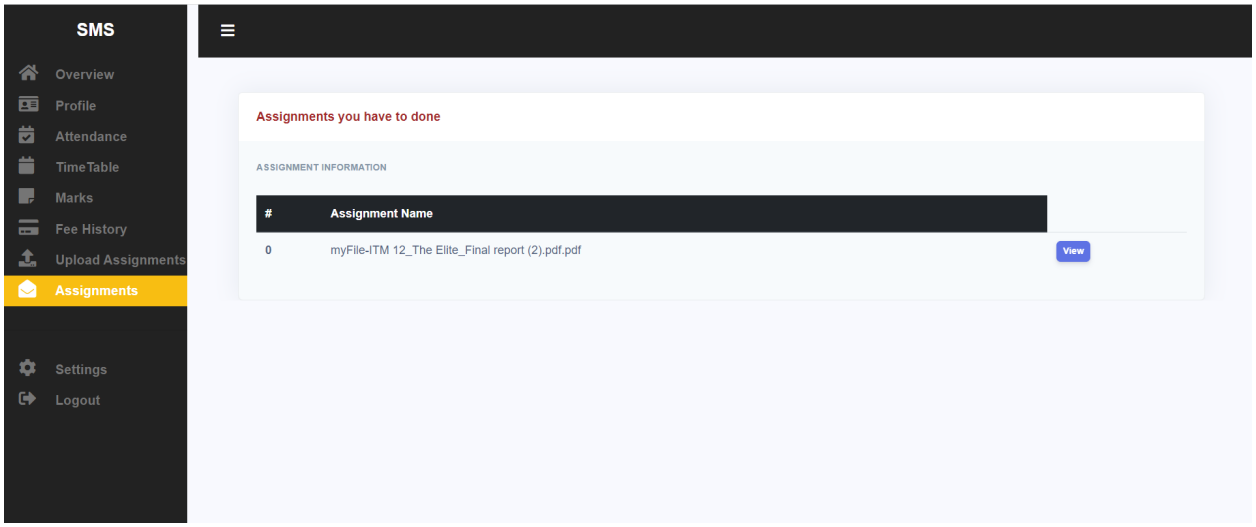


Figure 48: UI for student view assignments and homework

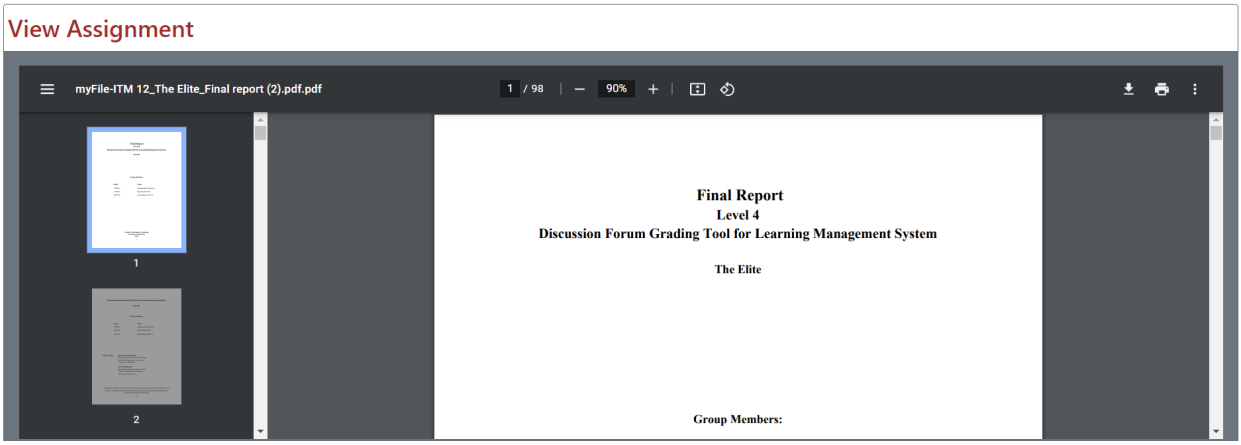


Figure 49: UI for student view and download assignments and homework

5.1.3. Parent Module

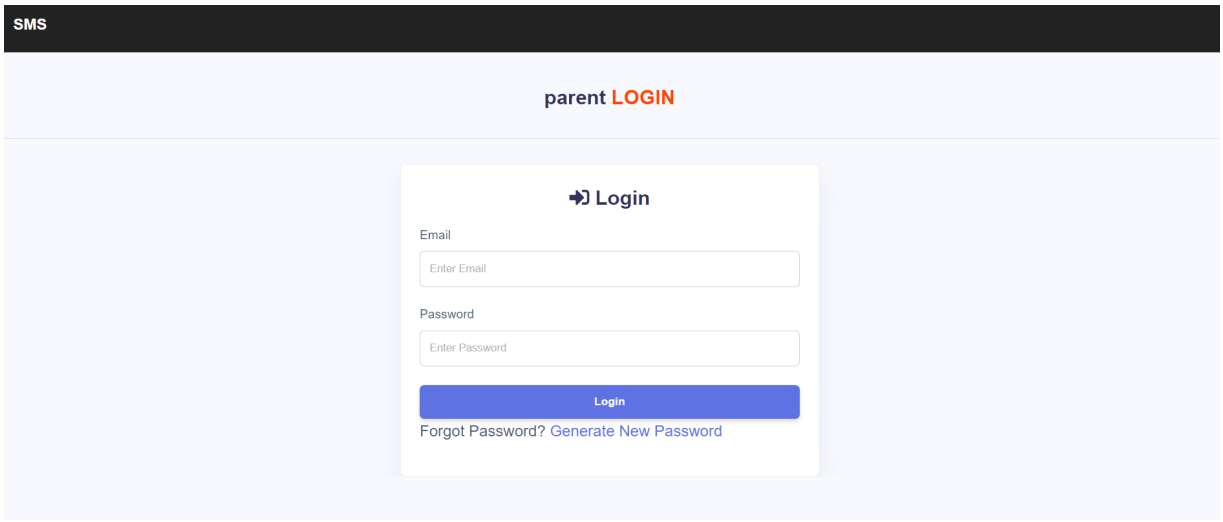


Figure 50: UI for parent login

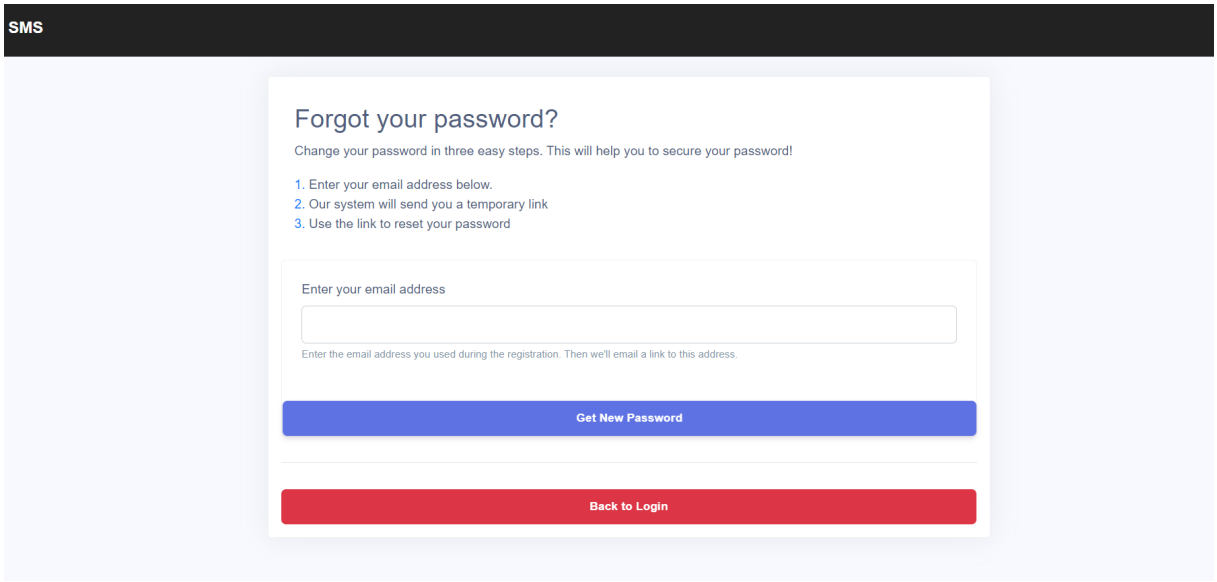


Figure 51: UI for parent forgot password

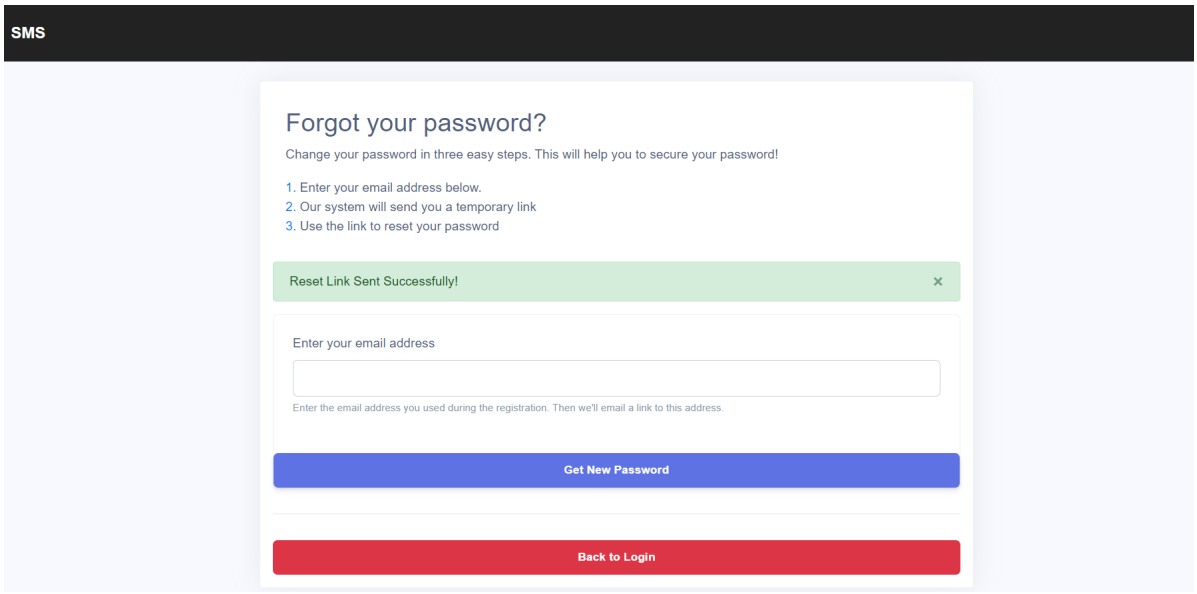


Figure 52: UI for parent password resetting link

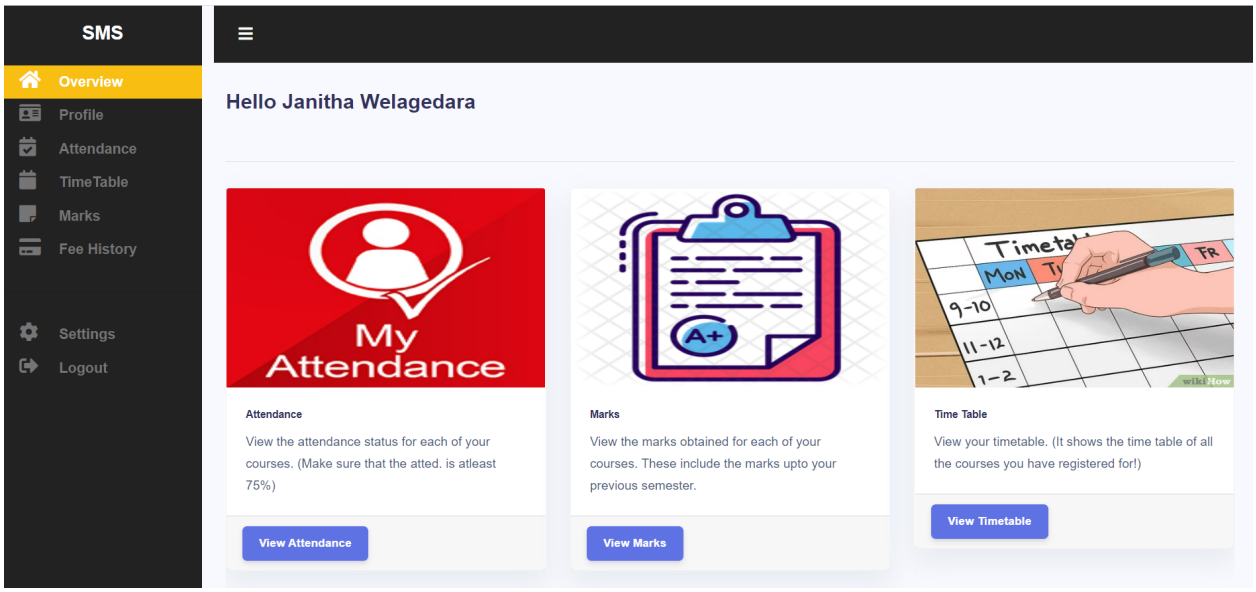


Figure 53: UI for parent overview

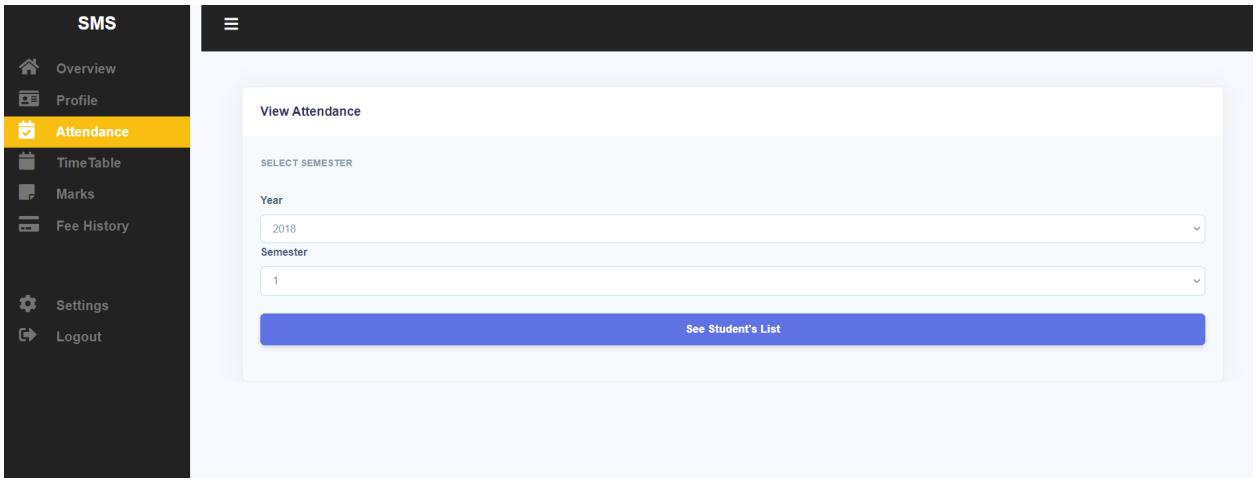


Figure 54: UI for view student's attendence

5.1.4. Teacher Module

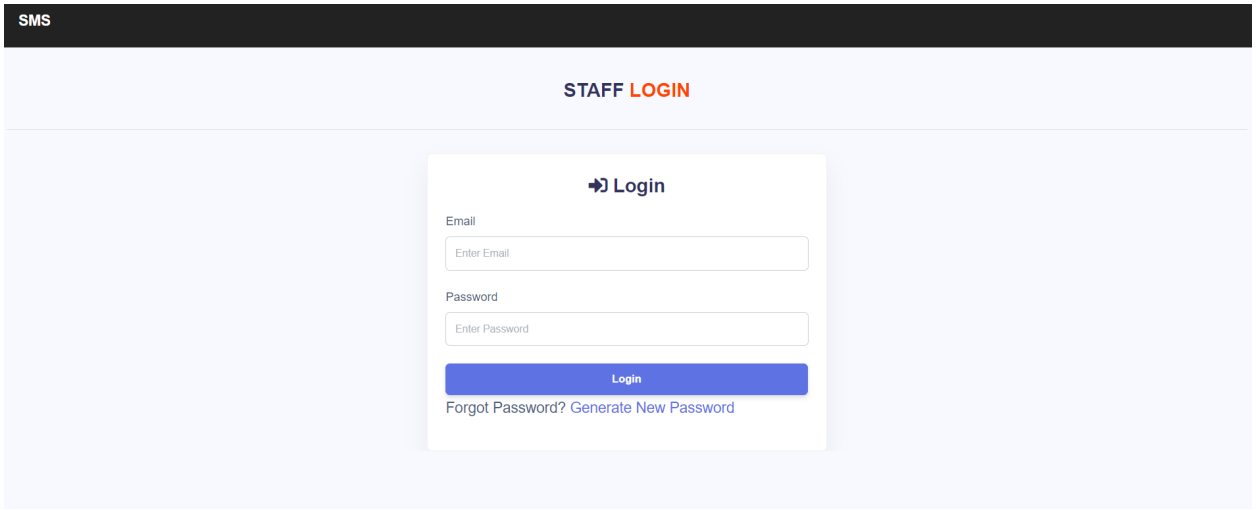


Figure 55: UI for teacher login

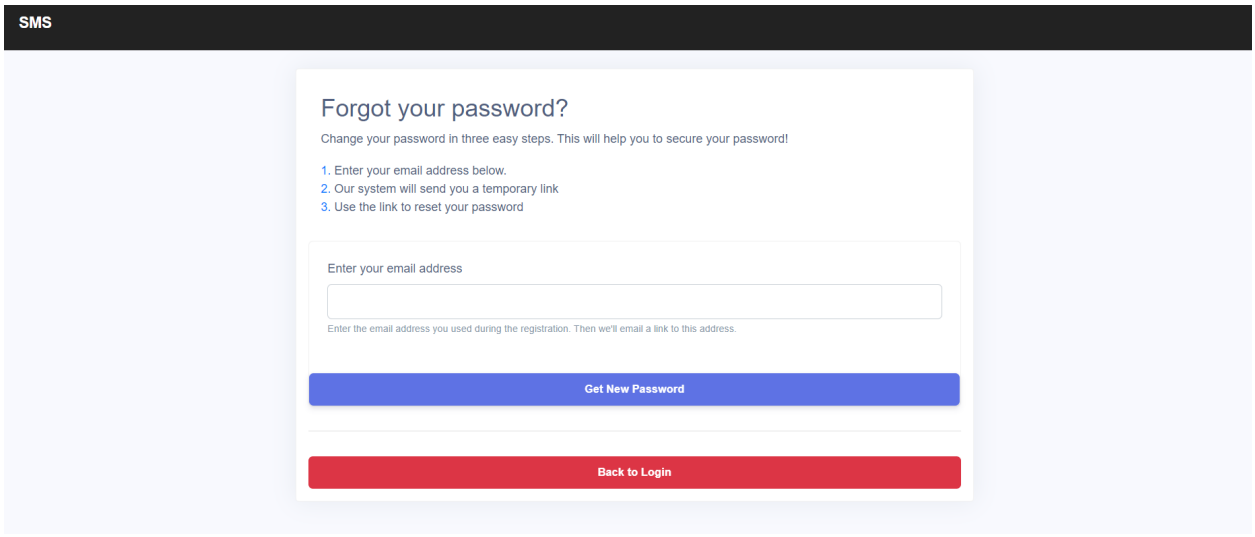


Figure 56: UI for teacher forgot password

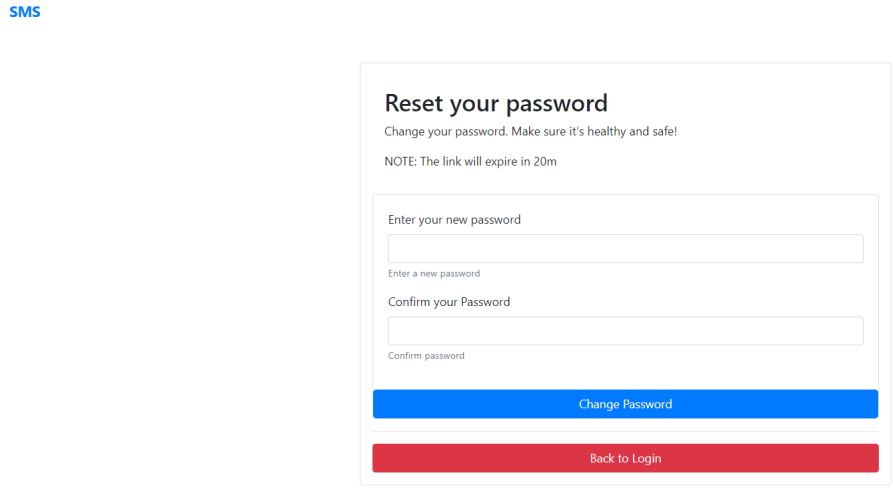


Figure 57: UI for teacher reset password

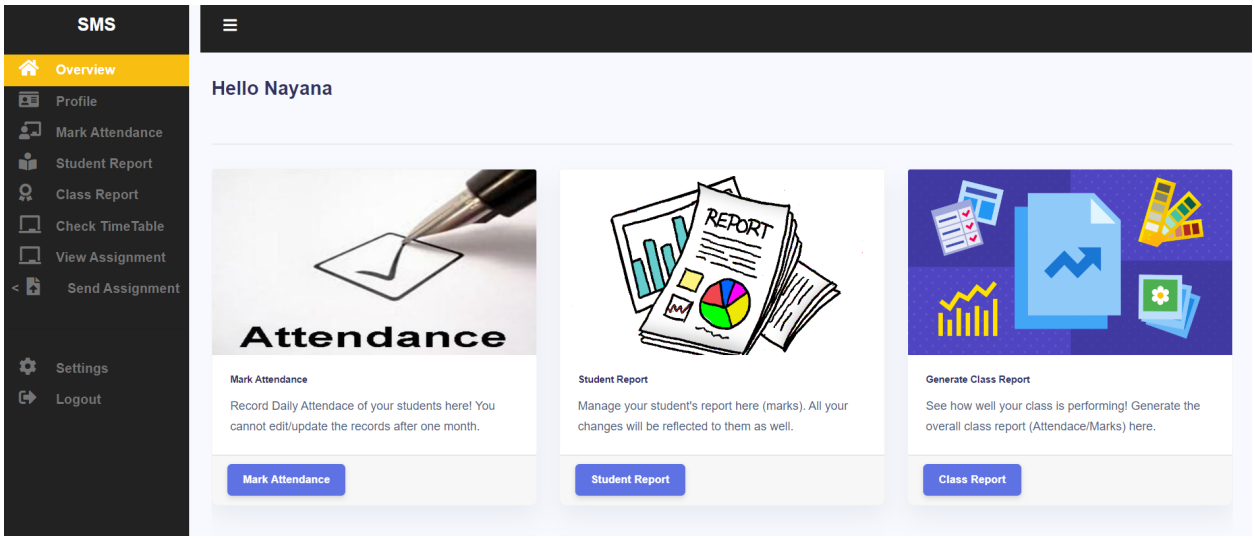


Figure 58: UI for teacher overview

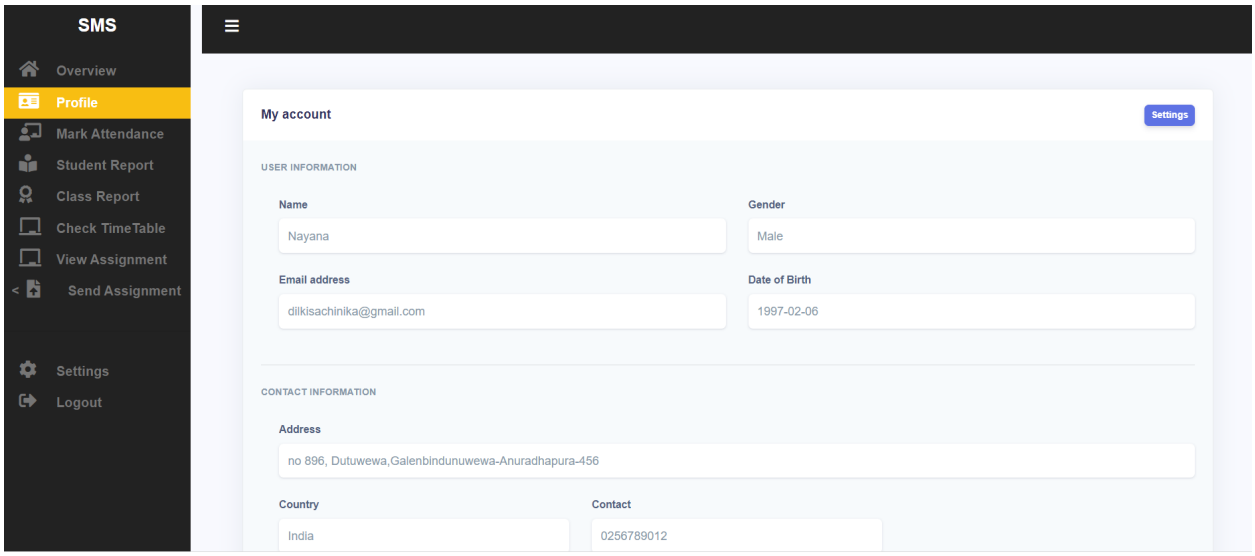


Figure 59: UI for teacher profile

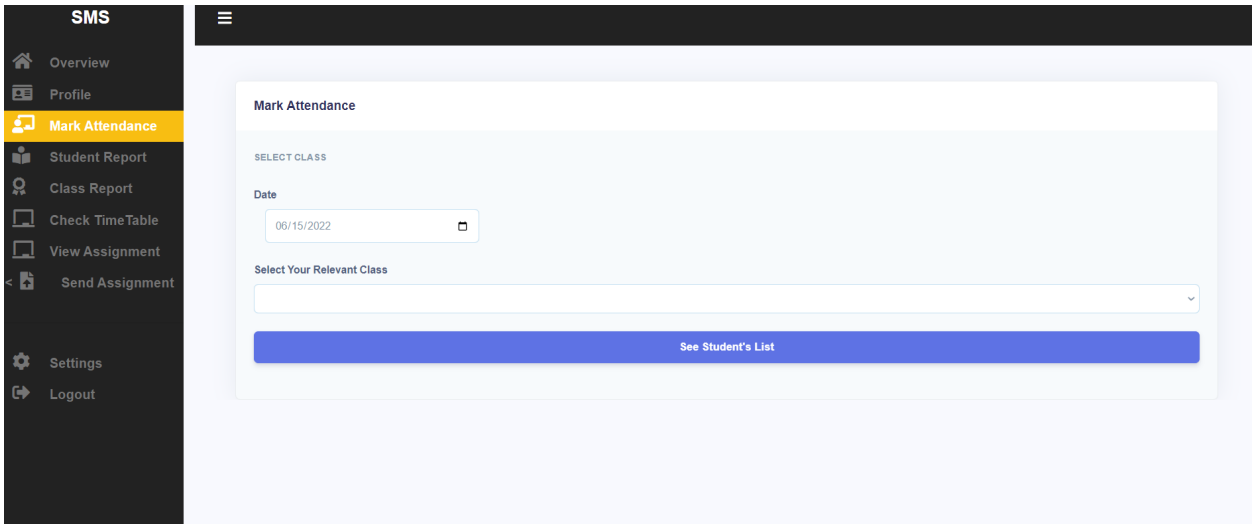


Figure 60: UI for teacher mark attendence

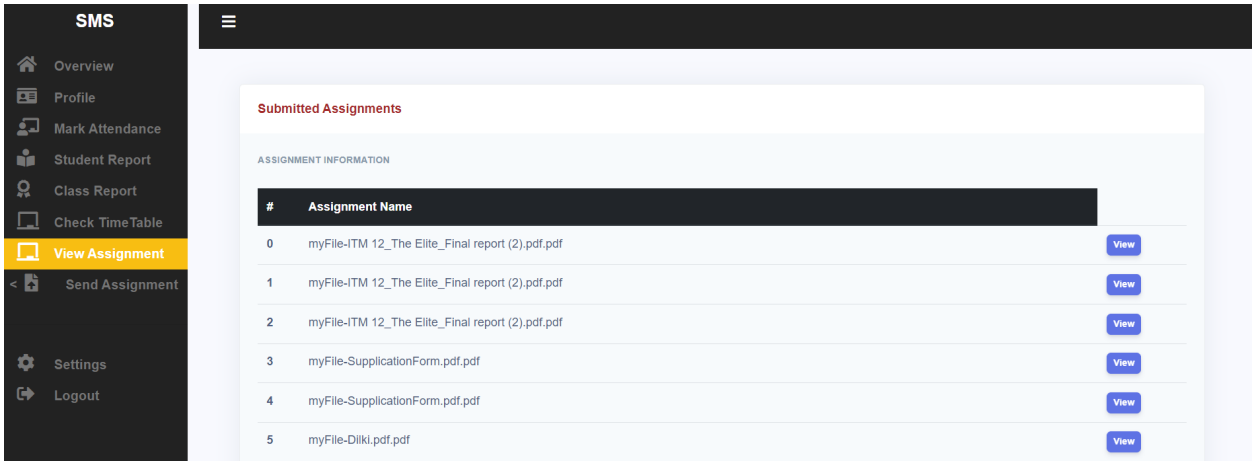


Figure 61: UI for teacher view submitted assignments

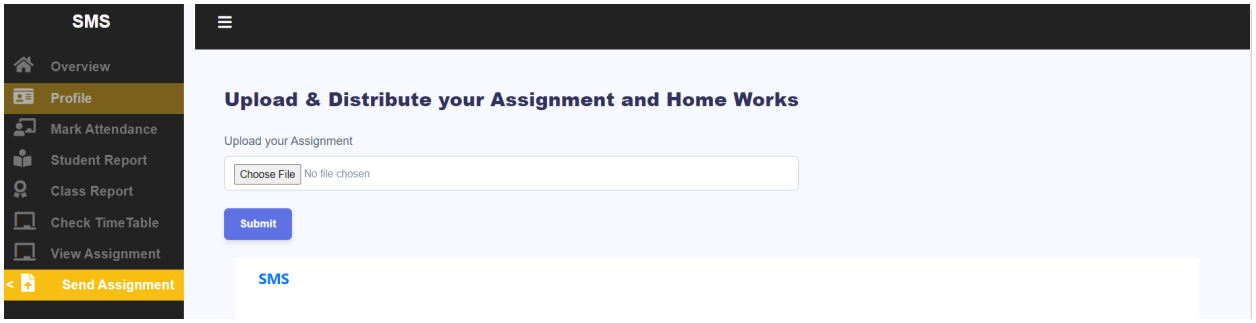


Figure 62: UI for teacher upload assignments and homework

5.1.2. Code segments

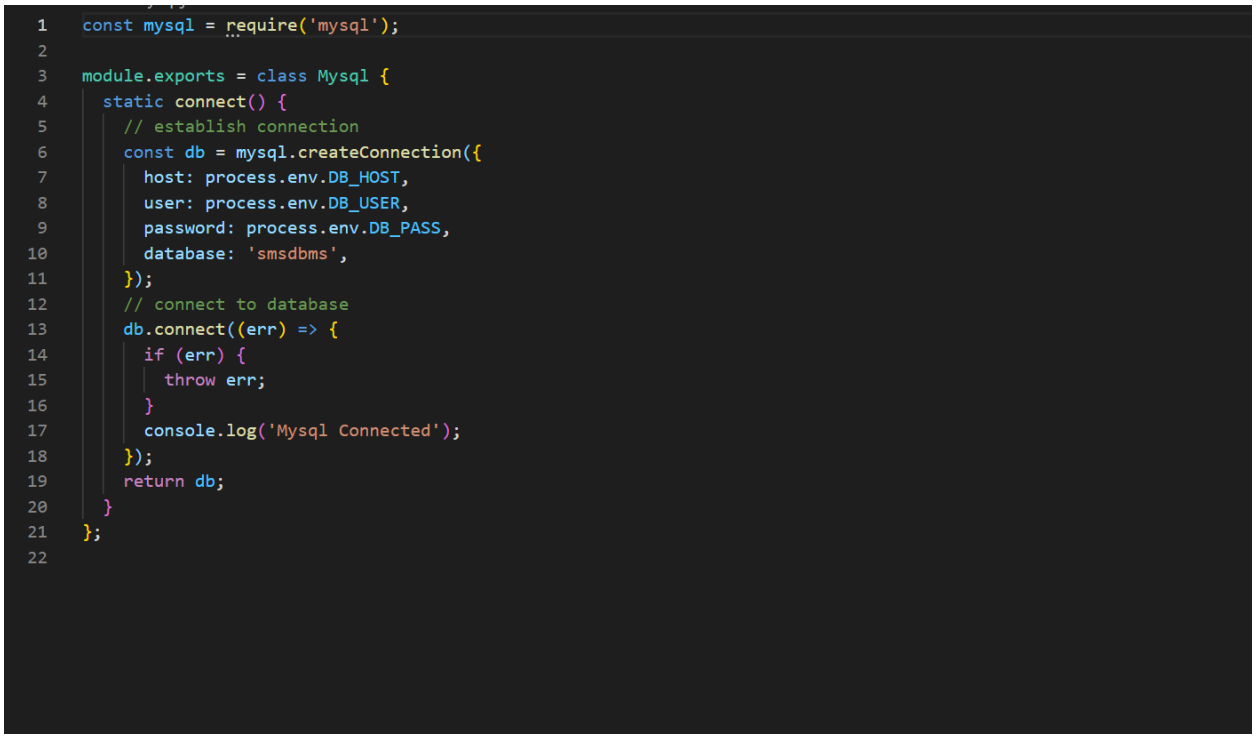
```sql
use smsdbms;

CREATE TABLE IF NOT EXISTS `admin`(
    `admin_id` VARCHAR(36) NOT NULL,
    `name` VARCHAR(255) NOT NULL,
    `email` VARCHAR(255) NOT NULL UNIQUE,
    `password` VARCHAR(255) NOT NULL,
    PRIMARY KEY(`admin_id`)
);

CREATE TABLE IF NOT EXISTS `course` (
    `c_id` VARCHAR(100) NOT NULL UNIQUE,
    `semester` INT NOT NULL,
    `name` VARCHAR(255) NOT NULL,
    `c_type` VARCHAR(255) NOT NULL,
    `credits` INT NOT NULL,
    `dept_id` VARCHAR(255) NOT NULL,
    PRIMARY KEY (`c_id`)
);

CREATE TABLE IF NOT EXISTS `student` (
    `s_id` VARCHAR(36) NOT NULL,
    `s_name` VARCHAR(255) NOT NULL,
    `gender` VARCHAR(6) NOT NULL,
    `dob` DATE NOT NULL,
    `email` VARCHAR(255) NOT NULL UNIQUE,
    `s_address` VARCHAR(255) NOT NULL,
    `contact` VARCHAR(12) NOT NULL,
    `password` VARCHAR(255) NOT NULL,
    `section` INT NOT NULL,
```

Figure 63: Code segment for the database



```
const mysql = require('mysql');

module.exports = class Mysql {
  static connect() {
    // establish connection
    const db = mysql.createConnection({
      host: process.env.DB_HOST,
      user: process.env.DB_USER,
      password: process.env.DB_PASS,
      database: 'smsdbms',
    });
    // connect to database
    db.connect((err) => {
      if (err) {
        throw err;
      }
      console.log('Mysql Connected');
    });
    return db;
  }
};
```

Figure 64: Code segment for the database

controllers > JS admin.js > ...

```
const mysql = require('mysql');
const jwt = require('jsonwebtoken');
const bcrypt = require('bcryptjs');
const uuidv4 = require('uuid').v4;
const mailgun = require('mailgun-js');
let ejs = require("ejs");
let pdf = require("html-pdf");
let path = require("path");
const fs = require("fs");
const DOMAIN = process.env.DOMAIN_NAME;
const mg = mailgun({apiKey: process.env.MAILGUN_API_KEY, domain: DOMAIN});

const db = mysql.createConnection({
    host: process.env.DB_HOST,
    user: process.env.DB_USER,
    password: process.env.DB_PASS,
    dateStrings: 'date',
    database: 'smsdbms',
});

// Students limit per section
const SECTION_LIMIT = 20;

// Database query promises
const zeroParamPromise = (sql) => {
    return new Promise((resolve, reject) => {
        db.query(sql, (err, results) => {
            if (err) return reject(err);
            return resolve(results);
        });
```

Figure 65: Code segment for the database

controllers > JS parent.js > ...

```
const mysql = require('mysql');
const jwt = require('jsonwebtoken');
const bcrypt = require('bcryptjs');

const mailgun = require('mailgun-js');
const DOMAIN = process.env.DOMAIN_NAME;
const mg = mailgun({ apiKey: process.env.MAILGUN_API_KEY, domain: DOMAIN });

const db = mysql.createConnection({
    host: process.env.DB_HOST,
    user: process.env.DB_USER,
    password: process.env.DB_PASS,
    dateStrings: 'date',
    database: 'smsdbms',
});

// Database query promises
const zeroParamPromise = (sql) => {
    return new Promise((resolve, reject) => {
        db.query(sql, (err, results) => {
            if (err) return reject(err);
            return resolve(results);
        });
    });
};

const queryParamPromise = (sql, queryParam) => {
    return new Promise((resolve, reject) => {
        db.query(sql, queryParam, (err, results) => {
            if (err) return reject(err);
```

Figure 66: Code segment controllers - Parent

controllers > JS staff.js > ...

```
const mysql = require('mysql');
const jwt = require('jsonwebtoken');
const bcrypt = require('bcryptjs');

const fs= require('fs');
const path=require('path');
// const dirPath= path.join(__dirname + '/public','doc');
// console.log("dirPath",dirPath);

const cur_dir = process.cwd();
const dirPath= path.join(cur_dir + '/public','doc');
let uploadFile;
// change current directory

// process.chdir('E:\\My_works\\17_batch\\new\\sms-backend');
// console.log("process.cwd() ",process.cwd());
//
// // path.join for cross-platform paths:
// process.chdir(path.join('E:', 'My_works', '17_batch'));
// console.log("process.cwd() 2 ",process.cwd());

const mailgun = require('mailgun-js');
const multer = require("multer");
const {v4: uuidv4} = require("uuid");
const DOMAIN = process.env.DOMAIN_NAME;
const mg = mailgun({ apiKey: process.env.MAILGUN_API_KEY, domain: DOMAIN });



const db = mysql.createConnection({
```

Figure 67: Code segment controllers - Staff

Figure 68: Code segment controllers - Student

middlewares > JS adminAuth.js > ...

```
const jwt = require('jsonwebtoken');
const mysql = require('mysql');

const db = mysql.createConnection({
  host: process.env.DB_HOST,
  user: process.env.DB_USER,
  password: process.env.DB_PASS,
  dateStrings: 'date',
  database: 'smsdbms',
});

const selectID = (id) => {
  return new Promise((resolve, reject) => {
    const sql1 = 'SELECT name FROM admin WHERE admin_id = ?';
    db.query(sql1, [id], (err, results) => {
      if (err) return reject(err);
      return resolve(results);
    });
  });
};

const requireAuth = (req, res, next) => {
  const token = req.cookies.jwt;
  if (token) {
    jwt.verify(token, process.env.JWT_SECRET, async (err, result) => {
      if (err) {
        req.flash(
          'error_msg',
          'You need to login as ADMIN in order to view that source!'
```

Figure 69: Code segment Authentication - Admin

middlewares > JS parentAuth.js > ...

```
const jwt = require('jsonwebtoken');
const mysql = require('mysql');

const db = mysql.createConnection({
    host: process.env.DB_HOST,
    user: process.env.DB_USER,
    password: process.env.DB_PASS,
    dateStrings: 'date',
    database: 'smsdbms',
});

const selectID = (id) => {
    return new Promise((resolve, reject) => {
        const sql1 = 'SELECT p_name FROM parent WHERE p_id = ?';
        db.query(sql1, [id], (err, results) => {
            if (err) return reject(err);
            return resolve(results);
        });
    });
};

const requireAuth = (req, res, next) => {
    const token = req.cookies.jwt;
    if (token) {
        jwt.verify(token, process.env.JWT_SECRET, async (err, result) => {
            if (err) {
                req.flash(
                    'error_msg',
                    'You need to login as PARENT in order to view that source! ==> '
                );
```

Figure 70: Code segment Authentication - Parent

middlewares > JS staffAuth.js > ...

```
const jwt = require('jsonwebtoken');
const mysql = require('mysql');

const db = mysql.createConnection({
  host: process.env.DB_HOST,
  user: process.env.DB_USER,
  password: process.env.DB_PASS,
  dateStrings: 'date',
  database: 'smsdbms',
});

const selectID = (id) => {
  return new Promise((resolve, reject) => {
    const sql1 = 'SELECT st_name FROM staff WHERE st_id = ?';
    db.query(sql1, [id], (err, results) => {
      if (err) return reject(err);
      return resolve(results);
    });
  });
};

const requireAuth = (req, res, next) => {
  const token = req.cookies.jwt;
  if (token) {
    jwt.verify(token, process.env.JWT_SECRET, async (err, result) => {
      if (err) {
        req.flash(
          'error_msg',
          'You need to login as STAFF in order to view that source!'
        );
```

Figure 71: Code segment Authentication - Staff

middlewares > JS studentAuth.js > ...

```
1   const jwt = require('jsonwebtoken');
2   const mysql = require('mysql');
3
4   const db = mysql.createConnection({
5     host: process.env.DB_HOST,
6     user: process.env.DB_USER,
7     password: process.env.DB_PASS,
8     dateStrings: 'date',
9     database: 'smsdbms',
10  });
11
12  const selectID = (id) => {
13    return new Promise((resolve, reject) => {
14      const sql1 = 'SELECT s_name FROM student WHERE s_id = ?';
15      db.query(sql1, [id], (err, results) => {
16        if (err) return reject(err);
17        return resolve(results);
18      });
19    });
20  };
21
22  const requireAuth = (req, res, next) => {
23    const token = req.cookies.jwt;
24    if (token) {
25      jwt.verify(token, process.env.JWT_SECRET, async (err, result) => {
26        if (err) {
27          req.flash(
28            'error_msg',
29            'You need to login as STUDENT in order to view that source! ==> '
30          );
```

Figure 72: Code segment Authentication - Student

6. 2 Summary

In this chapter we explained the implementations that we have done up to now. Currently we are developing the database and designing web and interfaces.

Chapter 6

Discussion and Conclusion

In here we hope to provide a software solution for a school to manage their day today activities which are done at the school.In system there are four roles.They are Admin,Teacher,Parent and Student.They have different kind of functionalities.As technologies we have used EJS,JavaScript,HTML and CSS. and Node JS. For the database we have used MySql.Through the proposed system the drawbacks in the current school management systems will be avoided while establishing a strong relationship among students, parents, teachers and school administration.

References

1.https://www.javacodegeeks.com/2016/05/limitations-monolithic-application-need-adaptingmicroservices-architecture.html
2.https://dzone.com/articles/why-microservices
3.https://articles.microservices.com/
4.https://www.aspitio.com/
5.https://spring.io/projects/spring-boot

Appendix A

Individual Contribution to the Project

175083J-Thilakarathna W.M.D.S.

I'm responsible to develop the student role in "School Management System". The student can log in to the system or register with the system. When he is login to the system, he can put the user's name and the password. Then the student can edit the profile also. When it comes to assignments and homework, when teacher is uploading assignments and homework, the student can view. As well as student can upload answers both assignments and home works. When teacher or admin add some kind of notices student can view those notices. Student is doing creative activities. Then student can upload those creative activities via the system. When teacher is uploading some quiz, the student can access to the quiz and do the quiz with the system and he can submit the quizzes via the system. As well as student can View Assignment marks, timetable, progress reports, attendance report and upcoming events in the school. I'm contributing to both backend and frontend for the "Student module". Rather than that I contributed to draw diagrams also.

175090D-Welagedara J.M.

I'm responsible to develop the parent role in "School Management System". They can log in to the system or register with the system. When parent is login to the system, he can put the user's name and the password. Then the student can edit the profile also. Parent can view the time table, view the marks of students, view the dashboard. As well as the parent can view the creative activities of the student. When the admin adds the progress report of the student, the parent can view the progress report also. The parent can view the attendance of the particular student. When the school is requesting some donations from parents, parent can do donations. As well as the parent can pay school fees. The parent can chat with the teacher. I'm contributing to both backend and frontend for the "Teacher module." Rather than that I contributed to draw diagrams also.

175064D-Prasandika K.A.C

As the member of the school management system project, first of all, we discussed with the client, and we got a software project. Before starting implementation, we divided our project into 4 modules. Then I assigned the teacher module. I draw activity diagrams and class diagrams, and the functionalities associated with my module are: marking attendance, uploading quizzes, uploading notices, uploading marks, etc. I learned about React and NodeJS and am continuing to work.

175053T-Nasrina M.I.F.

I am responsible for admin module in the system. I am contributing to design front end and backend part of admin module.

We together designed all the design diagrams, database and all the documentation works. The mainly assigned diagram for me was Class diagram and I developed it well and I developed activity diagrams to my module in our system. The functionalities assigned for me is Admin module such as: add profiles, upload time table and course information (subjects), Generating various type of report as attendance report, progress report, uploading notices view notices and handling payments. For now, I designed user interface of admin and I have finished the front end works for that module. In backend I have working for the admin module. Now I am continuing with the further works assigned for me.